Capstone Project Report MINT-709 Failover solution for Elastix server over WAN

Master of Science in Internetworking,
University of Alberta

Ketan Sharma

Acknowledgement

I owe a debt of gratitude to Dr. Mike MacGregor - Professor and Chair, department of computing science, University of Alberta for the vision and foresight which inspired me to conceive this project.

It is also my duty to record my thankfulness to Shahnawaz Mir, Program coordinator, who provided me this opportunity to work on this project. Without his support and appreciation, it was difficult to achieve this target.

I am grateful to Mr.WalterLuzuriaga, who worked as a bridge between me and Eduardo Abad- QA & Certifications PaloSanto Solutions, who provided me study material which played a vital role in getting the final results.

Table of Contents

Introduction to Elastix Server

Elastix is an Open Source Software to establish Unified Communications. Elastix goal is to incorporate all the communication alternatives, available at enterprise level into a unique solution. Elastix not only provides telephony, it integrates communication alternatives to make an organization's environment more productive and efficient. From the last century Telephony was the traditional way that leads communications, that's why many users are focusing on their requirements to establish telephony communications in their organizations.

Some of the basic Features of Elastix include:

- Voicemail
- Fax-to-email
- Support for soft phones
- Web Interface Configuration
- Virtual conference rooms
- Call recording
- Least Cost Routing
- Extension Roaming
- PBX Interconnection
- Caller ID
- CRM
- Advance Reports

Elastix is an open source entrepreneur tool that can be used for commercial or personal purposes subject to the conditions as described under it's license GPLv2.All the Elastix software are full version and free to be used by anyone under it's license.

Elastix include the following communication media:

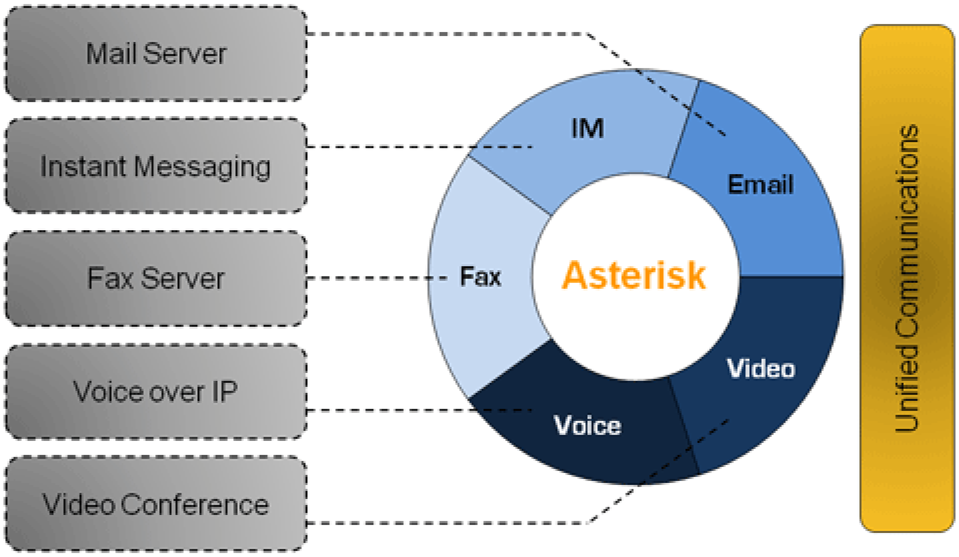


Figure 1

Ref: -http://Elastix.org/

Introduction to High Availability clustering with DRBD + Heartbeat

High availability clustering is a system design principle for increased availability. One of the most common implementations for HA is redundancy with failover. For example, we can define multiple paths to a given resource. High availability clustering is used with Heartbeat and Distributed Replicated Block Device (DRBD).

Introduction to DRBD and Heartbeat

The Distributed Replicated Block Device (DRBD) provides a networked version of data mirroring, classified under the redundant array of independent disks (RAID). DRBD effectively puts mirrored file system beneath the database. DRBD helps all the slave machines in replicating the changes which take place at any point of time on master server whose ip addresses are defined in drbd.conf file. We can define as many machines as we want as a backup server by giving them priority.

Heartbeat is a daemon which runs on all the machines in the high availability cluster. Heartbeat communicates with all the nodes which are participating in high availability cluster and decides the role of the nodes.

NODE 1 NODE 2

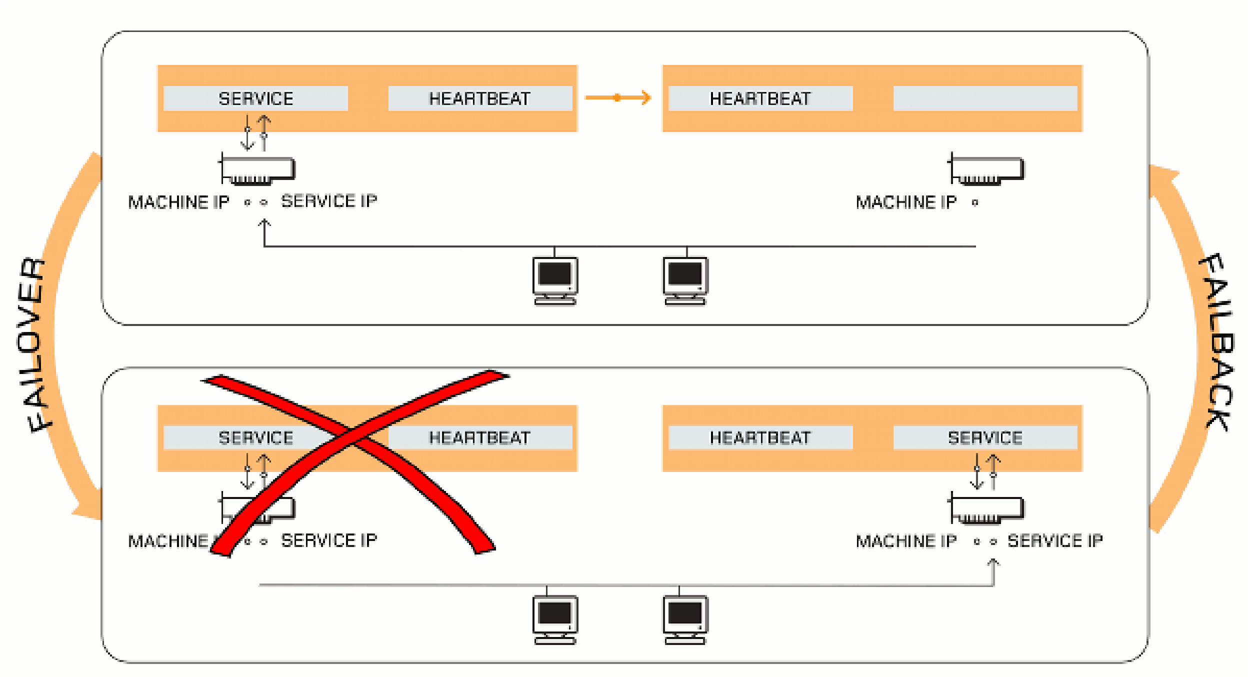


Figure 2

Ref: -http://www.drbd.org/home/what-is-ha/

The upper block in this image shows that the node 1 in the left hand side is active and the node 2 in the right hand side is passive. All the machines are getting their ip addresses from node 1 and there is heartbeat running between both of the nodes.

In the next block, as heartbeat stops between two nodes, node 2 is now acting as an active node and the machine in the left hand side is acting as standby. All the machines are getting their ip addresses from node 2. Services like ip addressing, mysql and http can be moved between the nodes without any change. Heartbeat is like a pulse between the nodes taking part in high availability clustering, which checks the status of the active server in regular intervals of time and when the heartbeat stops, either by system failure or because of administrative action there will be a failover/failback condition.

DRBD gives a flexibility of synchronizing data between the nodes from fully synchronous to asynchronous. There are three different replication modes supported by Distributed Replicated Block Device.

- **Protocol A**: It is an asynchronous replication protocol which is used for long distance replication scenarios. When this protocol is used with the combination of DRBD proxy it makes an effective recovery solution. As the local write operation is finished the replication packet is placed in TCP send buffer, but whenever there is forced failover then the data loss may occur. The data on the secondary servers is consistent. However the most recent changes before the crash can be lost.
- **Protocol B**: It is a semi-synchronous replication protocol used in DRBD. In this case there is no data loss encountered during forced failover, but if there occurs continues power failure on both the nodes and irreversible destruction of the primary nodes data happens then there will be data loss.
- **Protocol C**: It is a fully synchronous replication protocol. Data loss is inevitable even with this replication protocol if both nodes are irreversibly destroyed at the same time. This is because the local write operation is considered to be completed only when the write operation is confirmed on both the active and the passive nodes. Protocol C is the most commonly used protocol in DRBD because of its reliability.

Note: In this scenario we are using Protocol A.

Failover Solution
For Elastix Servers over Wide Area Network

We will start with step by step installation of Elastix Servers.

Note: HARD DRIVES ON BOTH SERVERS MUST HAVE EXACTLY THE SAME CAPACITY.

1. Boot from Elastix 2.0 latest stable version cd.

2. Choose default language for installation and for keyboard language.

3. Choose: Create custom Layout, for hard disk partition and erase all previous partitions.

4. Create: swap partition for 2048 MB size
   /boot for 100 MB with file system ext3
   / For 150 GB with file system ext3

5. Configure network interface and also configure manually the hostname for both the nodes.
   For Master: master.example.com (on both the servers)
   For Slave: slave.example.com

6. Then select the time zone and the root password for both the servers.

7. After the installation process server will reboot and ask for the root password for Mysql and for admin user which can be used for accessing the Elastix web interface of servers.

8. Execute from the shell:

```
yum update -y Elastix                    (on both the servers)
yum update -y
```

Note: this will update Elastix server to the latest stable version and add-ons.

Now we have to create the partition that will contain replication data on both the servers.

9. Fdisk /dev/sdb (in our scenario we are using second hard disk)
   Create a new partition (n)

Primary partition (p) (on both the servers)
Partition number (4) already selected by default
Press (w) this will write the partition and will exit.

10. Then reboot both the servers.

11. Define the file system for the new partition we have just created: sdb4

Command: mkfs.ext3 /dev/sdb4 (on both servers)

12. Now for synchronization type
Command**:** dd if=/dev/zero bs=1M count=1 of=/dev/sdb4; sync
(On both the servers)

13. We need to install the packages of drbd, heartbeat, openhpi

Command: yum install drbd83 kmod-drbd83 heartbeat openhpi -y
(On both the servers)

14. In this step we have to define the nodes on both servers and assign the ip addresses to the master and slave servers
Edit /etc/hosts

```
172.16.2.2 master.example.com
172.16.2.5 slave.example.com
```

15. Now we have to define the protocol and the locations of the servers which we want to be synchronized with each other.
Edit /etc/drbd.conf

```
        resource "r0" {
                protocol C;
                disk{on-io-error pass_on ;}
                startup{wfc-timeout 5; degr-wfc-timeout 3 ;}
                syncer{rate100M ;}
        on master.example.com {
                device /dev/drbd0;
        disk /dev/sdb4;
        address172.16.2.2:7789;
        meta-disk internal;
}
        on slave.example.com {
                device /dev/drbd0;
                disk /dev/sdb4;
                address 172.16.2.5:7789;
                meta-disk internal;   }}
```

16. Here we have to execute the following command on both the servers
    drbdadm create-md r0
    Then reboot both of the nodes.

17. Now execute
    service drbd start   (on both servers)
    For starting synchronization on both the servers, initially both the servers will be secondary and when we execute service drbd status.
    It will give:-
    Secondary/Secondary   (on both servers)

18. Execute drbdsetup /dev/drbd0 primary -o   (on master server only)
    For making one of the two servers a primary node.

19. We can determine the role of a server by executing the following command: drbdadm role r0

    The primary server should return: Primary/Secondary
    The secondary server should return: Secondary/Primary

20. We can mount the virtual partition /dev/drbd0, but first we must format the partition with ext3 format by using the following commands.
    mke2fs –j /dev/drbd0
    mkdir /replica
    mount /dev/drbd0 /replica   (only on master server)

21. Now we will make a directory named as replica only on secondary server
    mkdir /replica

22. Install Elastix call center module on both servers.
    yum install Elastix-callcenter -y

23. Only on Master server execute these commands

```
cd /replica

tar -zcvf etc-asterisk.tgz /etc/asterisk
tar -zcvf var-lib-asterisk.tgz /var/lib/asterisk
tar -zcvf var-www.tgz /var/www/
tar -zcvf var-lib-mysql.tgz /var/lib/mysql/
tar -zcvf opt-Elastix-dialer.tgz /opt/Elastix/dialer/
tar -zcvf usr-lib-asterisk.tgz /usr/lib/asterisk/

tar -zxvf etc-asterisk.tgz
tar -zxvf var-lib-asterisk.tgz
tar -zxvf var-www.tgz
tar -zxvf var-lib-mysql.tgz
tar -zxvf opt-Elastix-dialer.tgz
tar -zxvf usr-lib-asterisk.tgz

rm -rf etc-asterisk.tgz
rm -rf var-lib-asterisk.tgz
rm -rf var-www.tgz
rm -rf var-lib-mysql.tgz
rm -rf opt-Elastix-dialer.tgz
rm -rf usr-lib-asterisk.tgz

rm -rf /etc/asterisk
rm -rf /var/lib/asterisk
rm -rf /var/www/
rm -rf /var/lib/mysql/
rm -rf /opt/Elastix/dialer/
rm -rf /usr/lib/asterisk/

ln -s /replica/etc/asterisk/ /etc/asterisk
ln -s /replica/var/lib/asterisk/ /var/lib/asterisk
ln -s /replica/var/www/ /var/www
ln -s /replica/var/lib/mysql/ /var/lib/mysql
ln -s /replica/opt/Elastix/dialer/ /opt/Elastix/dialer
ln -s /replica/usr/lib/asterisk/ /usr/lib/asterisk
```

24. Only on slave server execute the following commands

```
rm -rf /etc/asterisk
rm -rf /var/lib/asterisk
rm -rf /var/www/
rm -rf /var/lib/mysql/
rm -rf /opt/Elastix/dialer/
rm -rf /usr/lib/asterisk/
ln -s /replica/etc/asterisk/ /etc/asterisk
ln -s /replica/var/lib/asterisk/ /var/lib/asterisk
ln -s /replica/var/www/ /var/www
ln -s /replica/var/lib/mysql/ /var/lib/mysql
ln -s /replica/opt/Elastix/dialer/ /opt/Elastix/dialer
ln -s /replica/usr/lib/asterisk/ /usr/lib/asterisk
```

25. In this step stop all the services that heartbeat can control. (on both the servers)

For that execute the following:

```
service asterisk stop
service mysqld stop
service httpd stop
service Elastixdialer stop
service dahdi stop
chkconfig asterisk off
chkconfig mysqld off
chkconfig httpd off
chkconfig Elastixdialer off
chkconfig dahdi off
```

Note: From here we are going to configure high availability clustering between the nodes.

There are three important files we need to take in consideration while configuring high availability clustering.

a. ha.cf
b. haresources

c. authkeys

26. First of all we have to create /etc/ha.d/ha.cf (on both the servers)

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility local0
keepalive 2
deadtime 20
warntime 10
initdead 40
udpport 694
bcast eth0 # Linux
auto_failback on
node master.example.com
node slave.example.com
```

Warn time issues a warning that a node is no longer available.

Dead time is the time after which Heartbeat considers a node confirmed dead.

Initdead is the maximum time a master server waits for other nodes to check in at cluster startup.

Keepalive sets the interval at which Heartbeat keep-alive packets are sent to other nodes.

The node option identifies cluster members.

27. Create /etc/ha.d/haresources (on both servers)

```
master.example.com drbddisk::r0 Filesystem::/dev/drbd0::/replica::ext3
IPaddr::192.168.1.3/24/eth0/172.16.2.10 dahdi asterisk mysqld httpd
Elastixdialer
```

(This is a single line), 172.16.2.10 will be the floating IP address.

All the services which we want to start are defined in this file. This file should be same on each and every node in the cluster.

28. Create /etc/ha.d/authkeys (on both servers)

```
auth 1
1 sha1 Elastix
```

We can use sha1 or md5 whichever we want for security and Elastix is the actual authentication key.

Till here our entire important file required for high availability clustering has been created on all of the nodes participating in the cluster.

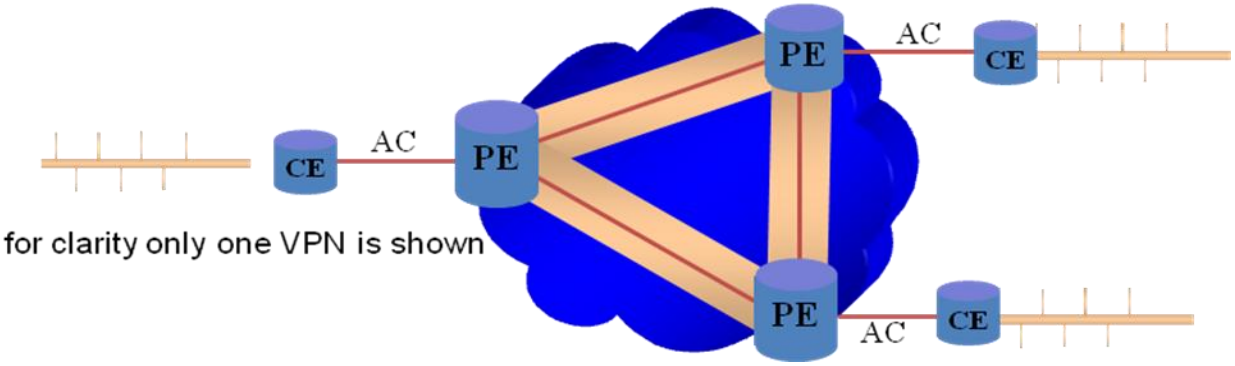
29. Execute the following commands:

```
chmod 600 /etc/ha.d/authkeys          (both servers)
chkconfig heartbeat on                (both servers)
cd /                              (only on the primary server)
umount /replica                   (only on the primary server)
service drbd restart                  (both servers)
drbdadm role r0                       (both servers)
Service heartbeat  restart            (both servers)
```

This configuration can work on local area network perfectly fine, where both the servers and phones are at the same location. But there is an issue with Elastix server that they can't work on different networks. We performed two different tests to check the behavior of the servers over Wide area network which are explained below.

Test 1

Elastix servers can only synchronize with each other within the cluster if, all the nodes are on the same network. Keeping this in mind we used VPLS (Virtual Private LAN Service). VPLS emulates a Local Area Network over an MPLS network, by connecting geographically dispersed sites through pseudo-wires. VPLS helps to share the same Ethernet broadcast domain between different sites.



CE is the Customer's Edge
PE is the Service Providers Edge

There will be a MPLS tunnel between each pair of PE's which is full mesh. In our test we have used two of the Alcatel-Lucent routers as PE's.

Below given is the network diagram we have used for VPLS connectivity between two servers.

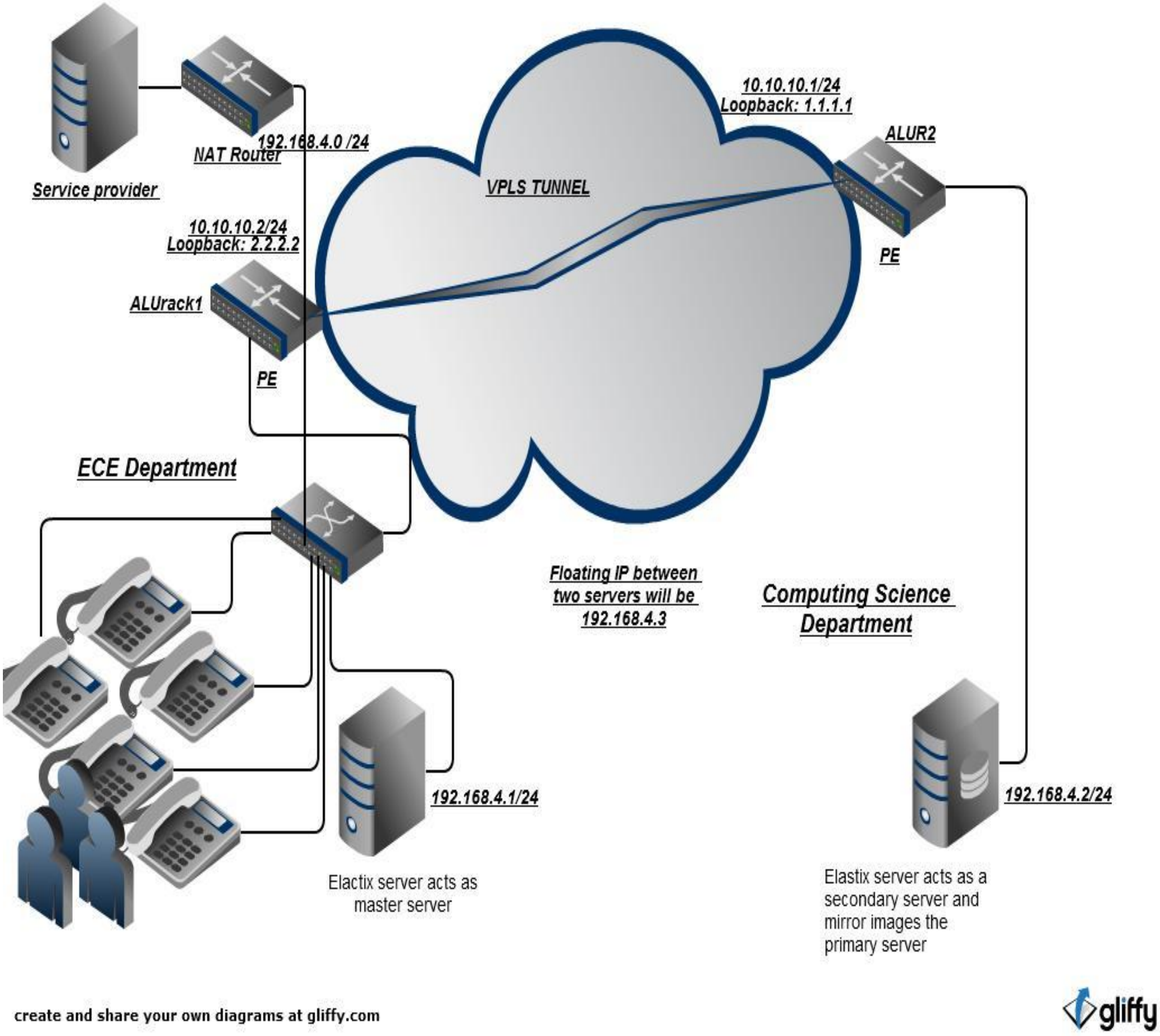

Service provider
NAT Router
192.168.4.0 /24
10.10.10.2/24
Loopback: 2.2.2.2
ALUrack1
PE
VPLS TUNNEL
10.10.10.1/24
Loopback: 1.1.1.1
ALUR2
PE
ECE Department
Floating IP between two servers will be 192.168.4.3
Computing Science Department
192.168.4.1/24
Elactix server acts as master server
192.168.4.2/24
Elastix server acts as a secondary server and mirror images the primary server
create and share your own diagrams at gliffy.com
gliffy

Figure 3

Configuration for PE1

```
#-------------------------------------------------

echo "System Configuration"

#-------------------------------------------------

system

name "PE1"

ccm 1

exit

#-------------------------------------------------

echo "Card Configuration"

#-------------------------------------------------

card 1

card-type iom-9g

mda 1

mda-type c1-1gb-sfp

exit

mda 2

mda-type c8-10/100eth-tx

exit

mda 4

mda-type c2-oc12/3-sfp

exit

exit
```

```
#--------------------------------------------------
echo "Port Configuration"
#--------------------------------------------------
port 1/2/1
ethernet
mtu 1536
exit
no shutdown
exit
port 1/2/2
ethernet
mode access
exit
no shutdown
exit
port 1/2/3
ethernet
exit
no shutdown
exit
#--------------------------------------------------
echo "Router (Network Side) Configuration"
#--------------------------------------------------
router
interface "system"
```

```
address 2.2.2.2/32

exit

interface "toALUR2"

address 10.10.10.2/24

port 1/2/1

exit

#------------------------------------------------

echo "OSPFv2 Configuration"

#------------------------------------------------

ospf 2.2.2.2

area 0.0.0.0

interface "system"

exit

interface "toALUR2"

exit

exit

exit

#------------------------------------------------

echo "LDP Configuration"

#------------------------------------------------

ldp

interface-parameters

interface "toALUR2"

exit

exit
```

```
targeted-session

exit

exit

exit


#-------------------------------------------------

echo "Service Configuration"

#-------------------------------------------------

service

customer 1 create

description "Default customer"

exit

sdp 12 mpls create

far-end 1.1.1.1

ldp

keep-alive

no shutdown

exit

no shutdown

exit

vpls 5 customer 1 create

stp

no shutdown

exit

sap 1/2/2 create
```

```
exit
mesh-sdp 12:5 create
exit
no shutdown
exit
exit
#--------------------------------------------------
echo "Router (Service Side) Configuration"
#--------------------------------------------------
router
#--------------------------------------------------
echo "OSPFv2 Configuration"
#--------------------------------------------------
ospf 2.2.2.2
exit
exit
exit all
# Finished SUN FEB 05 21:16:04 2012 UTC
```

Configuration for PE2

```
#--------------------------------------------------
echo "System Configuration"
#--------------------------------------------------
```

```
system
name "PE2"
ccm 1
exit
#--------------------------------------------------
echo "Card Configuration"
#--------------------------------------------------
card 1
card-type iom-9g
mda 1
mda-type c1-1gb-sfp
exit
mda 2
mda-type c8-10/100eth-tx
exit
mda 4
mda-type c2-oc12/3-sfp
exit
exit
#--------------------------------------------------
echo "Port Configuration"
#--------------------------------------------------
port 1/2/1
ethernet
mtu 1536
```

```
exit
no shutdown
exit
port 1/2/2
ethernet
mode access
exit
no shutdown
exit
port 1/2/3
ethernet
exit
no shutdown
exit
exit
#--------------------------------------------------
echo "Router (Network Side) Configuration"
#--------------------------------------------------
router
interface "system"
address 1.1.1.1/32
exit
interface "toALUrack1"
address 10.10.10.1/24
port 1/2/1
```

```
exit

#----------------------------------------------------

echo "OSPFv2 Configuration"

#----------------------------------------------------

ospf 1.1.1.1

area 0.0.0.0

interface "system"

exit

interface "toALUrack1"

exit

exit

exit

#----------------------------------------------------

echo "LDP Configuration"

#----------------------------------------------------

ldp

interface-parameters

interface "toALUrack1"

exit

exit

targeted-session

exit

exit

exit
```

```
#--------------------------------------------------
echo "Service Configuration"
#--------------------------------------------------
service
customer 1 create
description "VoIP"
exit
sdp 12 mpls create
far-end 2.2.2.2
ldp
keep-alive
no shutdown
exit
no shutdown
exit
vpls 5 customer 1 create
stp
no shutdown
exit
sap 1/2/2 create
exit
mesh-sdp 12:5 create
exit
no shutdown
exit
```

exit

#---

echo "OSPFv2 Configuration"

#---

ospf 1.1.1.1

exit

exit

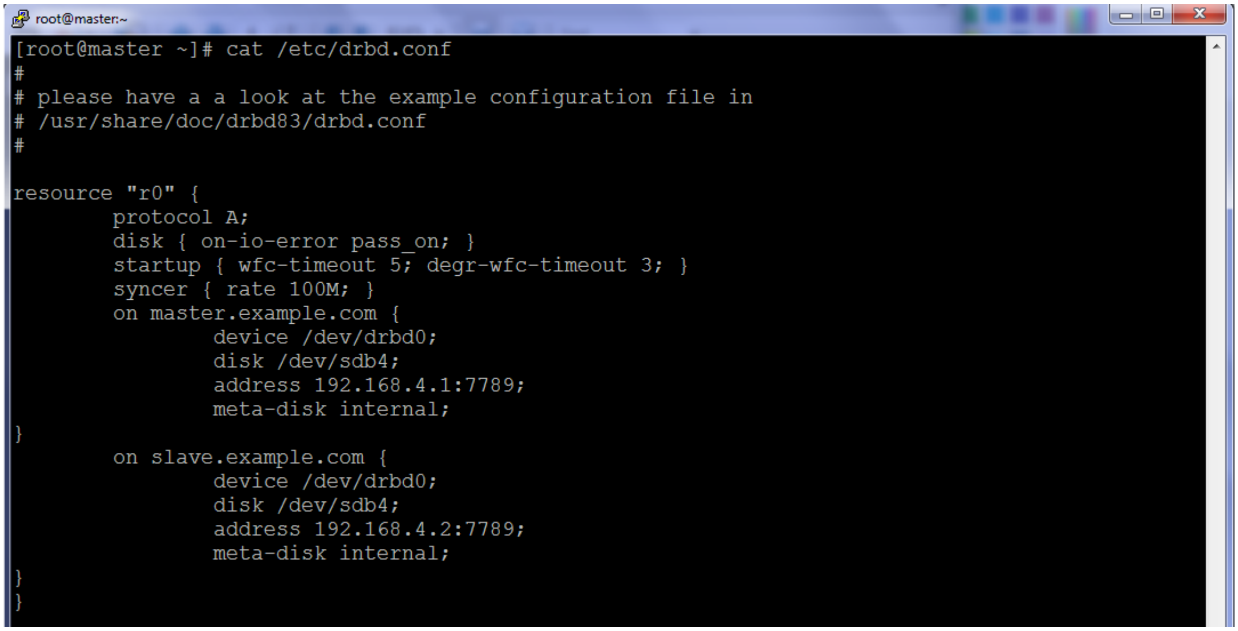
exit

Now there is VPLS tunnel running perfectly fine between Router PE1 and PE2.

DRBD configuration on master/slave server

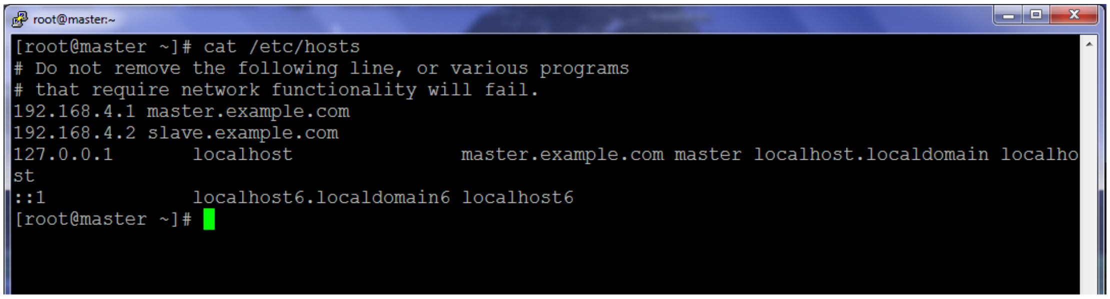
This file defines the location of the memory to be synchronized and the ip address of the master and slave nodes.

Note: This should be same on all the nodes within the cluster.



```
[root@master ~]# cat /etc/drbd.conf
#
# please have a a look at the example configuration file in
# /usr/share/doc/drbd83/drbd.conf
#

resource "r0" {
        protocol A;
        disk { on-io-error pass_on; }
        startup { wfc-timeout 5; degr-wfc-timeout 3; }
        syncer { rate 100M; }
        on master.example.com {
                device /dev/drbd0;
                disk /dev/sdb4;
                address 192.168.4.1:7789;
                meta-disk internal;
}
        on slave.example.com {
                device /dev/drbd0;
                disk /dev/sdb4;
                address 192.168.4.2:7789;
                meta-disk internal;
}
}
```

Here we are defining the host names of all the nodes in the cluster, which can communicate with each other. It should be same on all the nodes within the cluster.

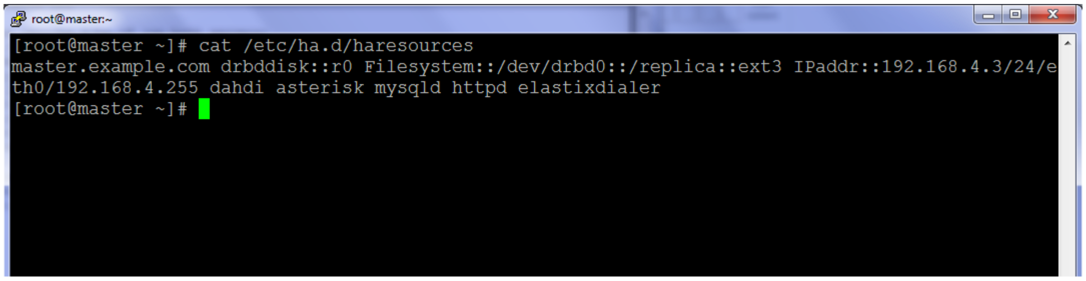


```
[root@master ~]# cat /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
192.168.4.1 master.example.com
192.168.4.2 slave.example.com
127.0.0.1       localhost               master.example.com master localhost.localdomain localho
st
::1             localhost6.localdomain6 localhost6
[root@master ~]#
```

These are the three most important files for the formation of high availability cluster.

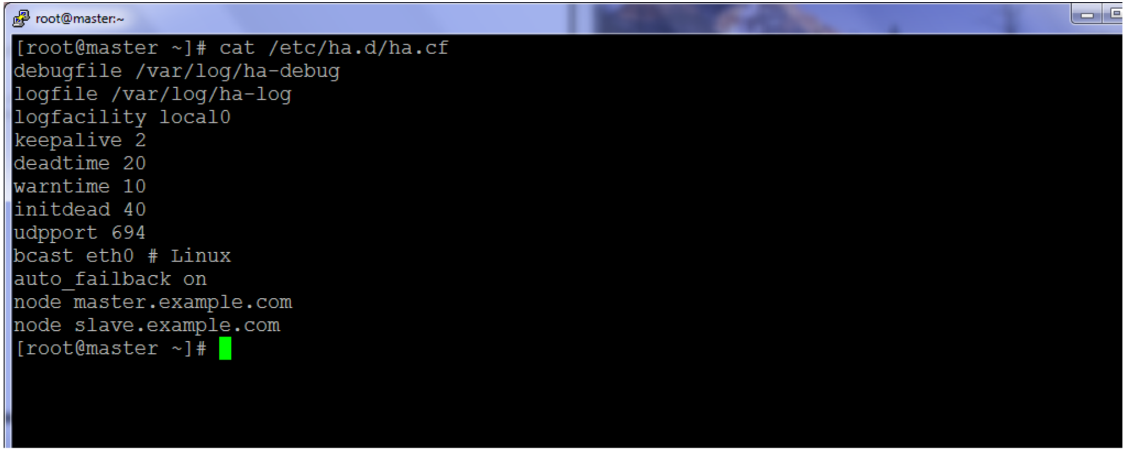
1. /etc/ha.d/haresources is a file where we have to define all the services which we want to run within the cluster. Floating ip address is also defined here.

NOTE: Everything written in this file should be in single line and should be the same on all the nodes otherwise it will not work.



```
[root@master ~]# cat /etc/ha.d/haresources
master.example.com drbddisk::r0 Filesystem::/dev/drbd0::/replica::ext3 IPaddr::192.168.4.3/24/e
th0/192.168.4.255 dahdi asterisk mysqld httpd elastixdialer
[root@master ~]#
```

2. As explained earlier in this file we will define keep alive, dead time, warn time, on the nodes which are participating. We can also declare auto failback condition as per our
requirements.



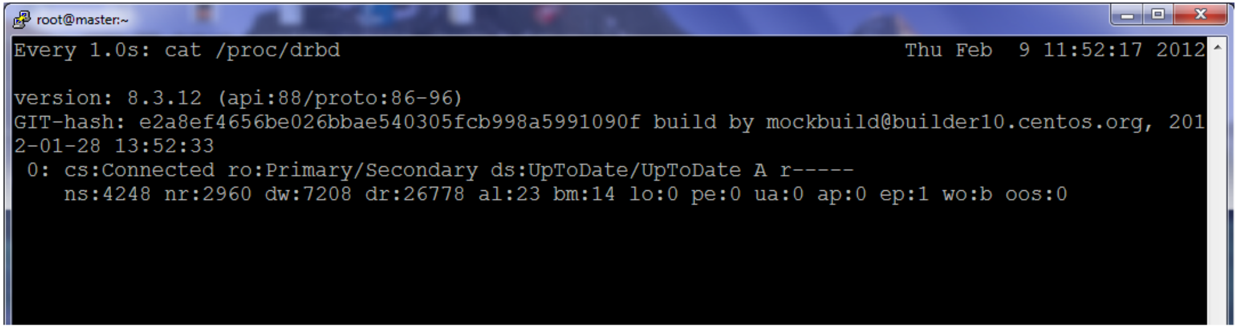
```
[root@master ~]# cat /etc/ha.d/ha.cf
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility local0
keepalive 2
deadtime 20
warntime 10
initdead 40
udpport 694
bcast eth0 # Linux
auto_failback on
node master.example.com
node slave.example.com
[root@master ~]#
```

Status and behavior of servers

Now it's time to check the synchronization status between two servers:

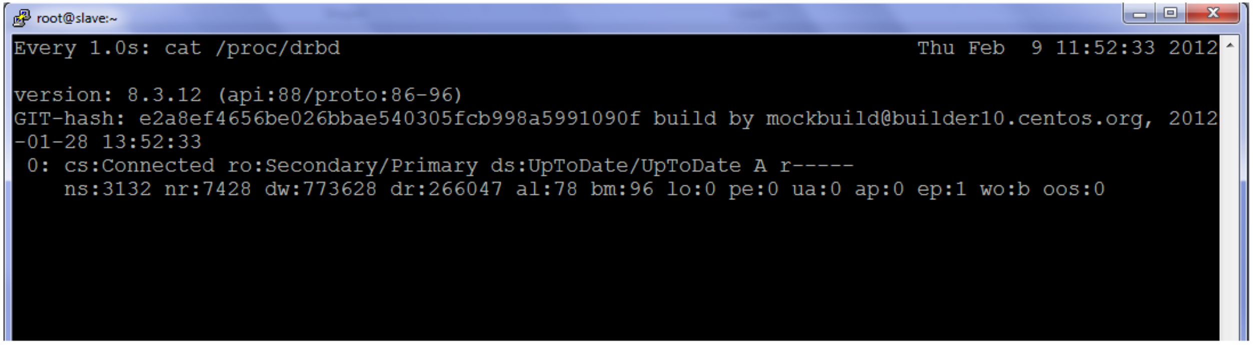
On Master/Slave servers: watch -n 1 cat /proc/drbd

Masters output:-



```
Every 1.0s: cat /proc/drbd                                        Thu Feb  9 11:52:17 2012

version: 8.3.12 (api:88/proto:86-96)
GIT-hash: e2a8ef4656be026bbae540305fcb998a5991090f build by mockbuild@builder10.centos.org, 201
2-01-28 13:52:33
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate A r-----
    ns:4248 nr:2960 dw:7208 dr:26778 al:23 bm:14 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
```

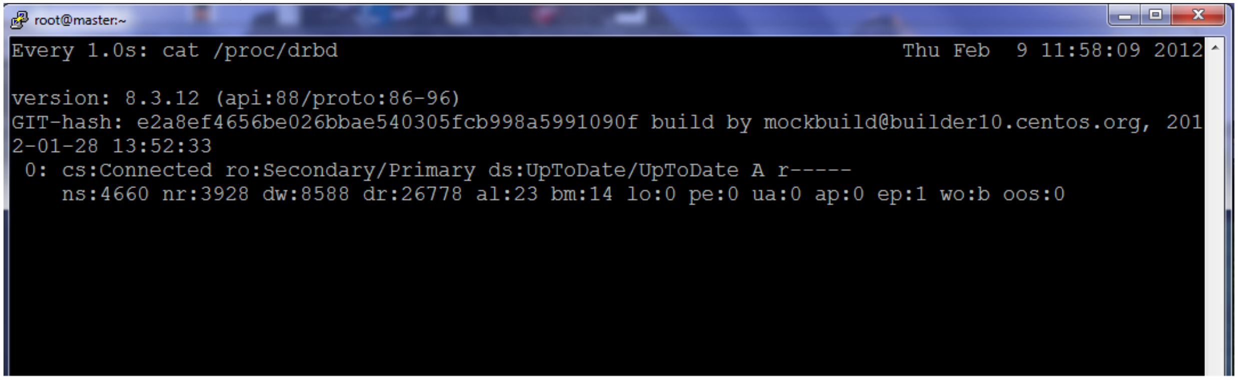
Slaves output:-



```
Every 1.0s: cat /proc/drbd                                        Thu Feb  9 11:52:33 2012

version: 8.3.12 (api:88/proto:86-96)
GIT-hash: e2a8ef4656be026bbae540305fcb998a5991090f build by mockbuild@builder10.centos.org, 2012
-01-28 13:52:33
 0: cs:Connected ro:Secondary/Primary ds:UpToDate/UpToDate A r-----
    ns:3132 nr:7428 dw:773628 dr:266047 al:78 bm:96 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
```

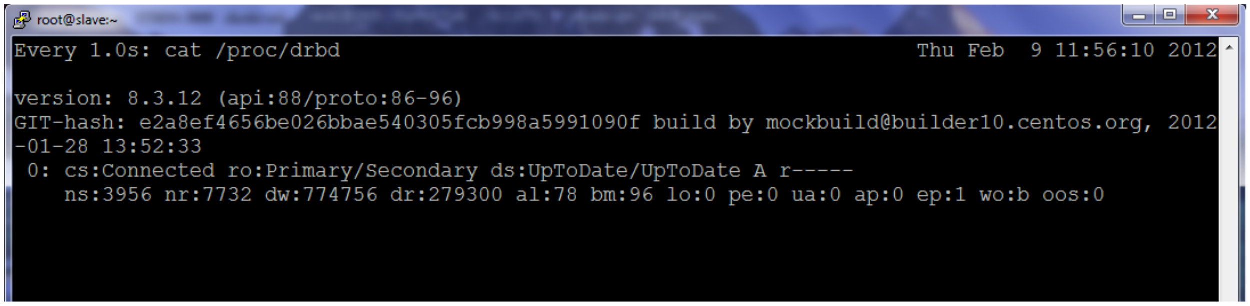
This shows that both the servers are connected and are up to date.

Command: service heartbeat stop (on master server, manually stopping the heartbeat)



```
Every 1.0s: cat /proc/drbd                                        Thu Feb  9 11:58:09 2012

version: 8.3.12 (api:88/proto:86-96)
GIT-hash: e2a8ef4656be026bbae540305fcb998a5991090f build by mockbuild@builder10.centos.org, 201
2-01-28 13:52:33
 0: cs:Connected ro:Secondary/Primary ds:UpToDate/UpToDate A r-----
    ns:4660 nr:3928 dw:8588 dr:26778 al:23 bm:14 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
```

When heart beat stops on master server our master server becomes secondary server and the slave server starts acting like a primary server as shown below and all the phones get registered with the secondary server with the help of floating ip address.



```
Every 1.0s: cat /proc/drbd                                    Thu Feb  9 11:56:10 2012

version: 8.3.12 (api:88/proto:86-96)
GIT-hash: e2a8ef4656be026bbae540305fcb998a5991090f build by mockbuild@builder10.centos.org, 2012
-01-28 13:52:33
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate A r-----
    ns:3956 nr:7732 dw:774756 dr:279300 al:78 bm:96 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
```

In our first test everything is working fine, both the servers are fully synchronized with the use of VPLS tunneling, but VPLS comes with cost. This is a service provided by the service providers, that's why we used Layer 2 Tunneling protocol v3 (L2TPv3) in our next test.

TEST 2

We are going to use Layer 2 Tunneling protocol v3 (L2TPv3) for the communication between two servers in this test. L2TPv3 provides high-speed Layer 2 tunneling or VPN services to end-user. L2TPv3 tunnels are available with the IOS basic IP package. It has combined the best features of two existing tunneling protocols: Cisco's Layer 2 Forwarding (L2F) and Microsoft's Point-to-Point Tunneling Protocol (PPTP). L2TPv3 corresponds to the data-link layer and use frames as its unit of exchange. L2TP operate on layer 2 and encapsulates the payload in a PPP frame to be sent across an internetwork.

Equipment used in Test 2

- 3 Cisco 2600 Routers
- 2 Cisco 2900 Routers
- 2 Cisco Catalyst 3750G switches
- 2 Sun Blade servers with Elastix 2.0 installed on them
- 2 IP Phones

Network Diagram for test 2.

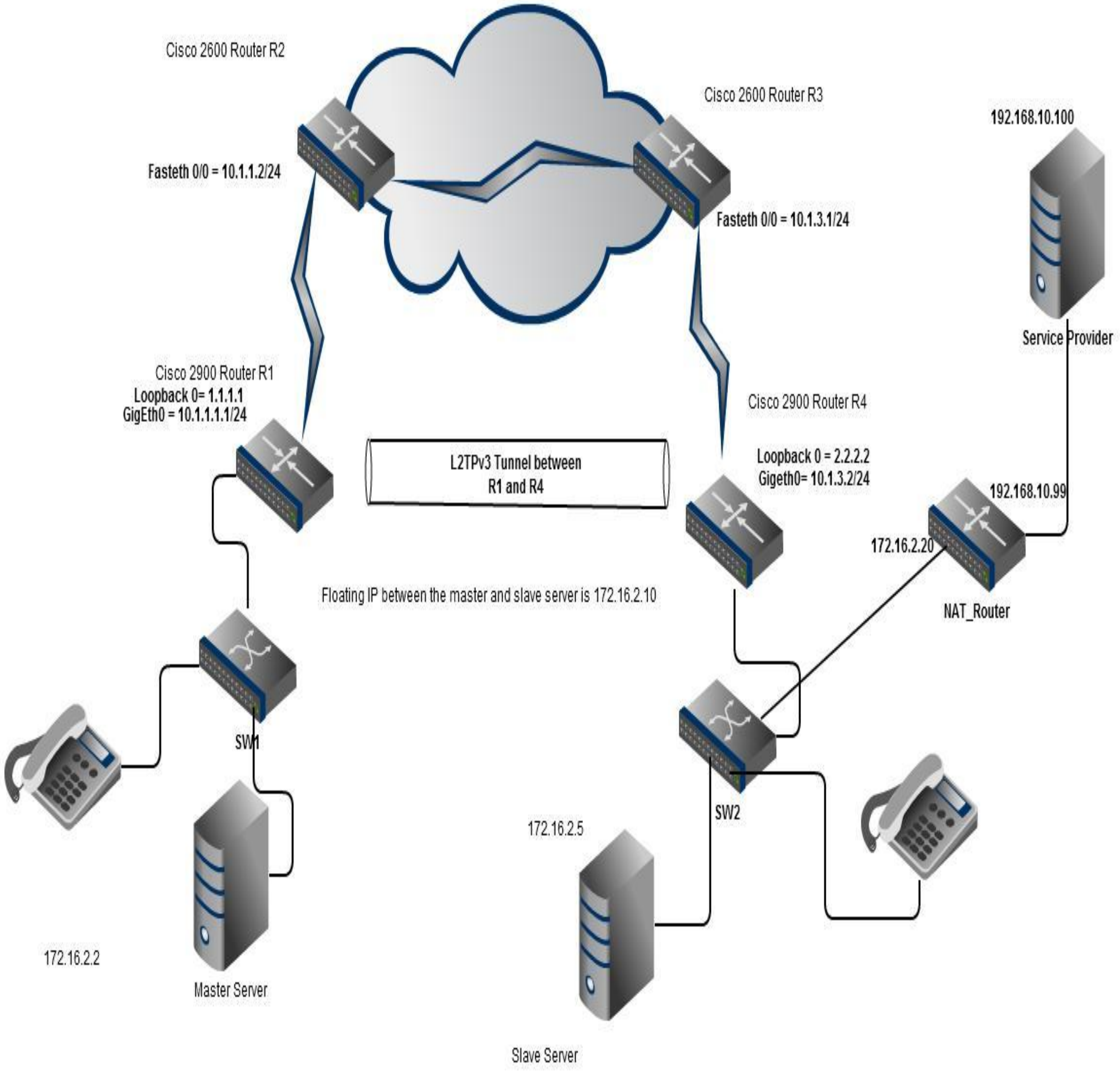


Figure 4

Configuration for Router R1

```
hostname R1
!
!
pseudowire-class MYPW
encapsulation l2tpv3
ip local interface Loopback0
!
!
!
interface Loopback0
ip address 1.1.1.1 255.255.255.255
 !
!
interface GigabitEthernet0/0
ip address 10.1.1.1 255.255.255.0
duplex auto
speed auto
 !
!
interface GigabitEthernet0/1
no ip address
duplex auto
speed auto
nokeepalive
xconnect 2.2.2.2 1 pw-class MYPW
 !
!
interface GigabitEthernet0/2
noip address
shutdown
duplex auto
speed auto
!
!
routereigrp 100
network 0.0.0.0
!
```

```
ip forward-protocol nd
```

Configuration for Router R2

```
hostname R2
!
!
interface FastEthernet0/0
ip address 10.1.1.2 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
noip address
!
interface Serial0/1
ip address 10.1.2.1 255.255.255.0
clock rate 8000000
!
routereigrp 100
network 0.0.0.0
no auto-summary
!
ip http server
ip classless
!
!
voice-port 1/0/0
!
voice-port 1/0/1
!
!
line con 0
line aux 0
linevty 0 4
!
!
end
```

Configuration for Router R3

```
hostname R3
!
!
!
interface FastEthernet0/0
ip address 10.1.3.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.1.2.2 255.255.255.0
!
interface FastEthernet0/1
noip address
shutdown
duplex auto
speed auto
!
interface Serial0/1
noip address
encapsulationppp
!
routereigrp 100
network 0.0.0.0
no auto-summary
!
voice-port 1/0/0
!
voice-port 1/0/1
!
!
line con 0
line aux 0
linevty 0 4
!
end
```

Configuration for Router R4

```
hostname R4
!
pseudowire-class MYPW
encapsulation l2tpv3
ip local interface Loopback0
!
interface Loopback0
ip address 2.2.2.2 255.255.255.255
!
interface GigabitEthernet0/0
ip address 10.1.3.2 255.255.255.0
duplex auto
speed auto
 !
!
interface GigabitEthernet0/1
noip address
duplex auto
speed auto
nokeepalive
xconnect 1.1.1.1 1 pw-class MYPW
 !
!
interface GigabitEthernet0/2
noip address
shutdown
duplex auto
speed auto
 !
routereigrp 100
network 0.0.0.0
!
ip forward-protocol nd
!
```

L2TP is configured between Router R1 and Router R4, as shown in the network diagram.

Configuration for Switch SW1

```
hostname SW1
!
!
interface GigabitEthernet1/0/1
switchport mode access
!
interface GigabitEthernet1/0/2
switchport mode access
!
interface GigabitEthernet1/0/3
switchport access vlan 10
!
interface GigabitEthernet1/0/6
switchport access vlan 10
switchport mode access
!
interface GigabitEthernet1/0/23
speed 100
duplex full
!
!
interface Vlan10
ip address 172.16.2.4 255.255.255.0
noip route-cache
standby 10 ip 172.16.2.1
standby 10 timers 2 4
standby 10 preempt
!
ip default-gateway 192.168.1.1
ip classless
ip http server
!
!
!
!
control-plane
!
!
```

```
line con 0
linevty 0 4
login
linevty 5 15
login
!
end
```

Configuration for Switch SW2

```
SW2#sh run
Building configuration...

Current configuration : 2082 bytes
!
version 12.2
no service pad
service timestamps debug datetimemsec
service timestamps log datetimemsec
no service password-encryption
!
hostname SW2
!
interface GigabitEthernet1/0/4
switchport trunk encapsulation dot1q
switchport mode trunk
!
!
interface GigabitEthernet1/0/5
switchport access vlan 10
switchport mode access
!
interface GigabitEthernet1/0/6
switchport access vlan 10
switchport mode access
!
!
interface GigabitEthernet1/0/8
switchport access vlan 10
!
!
interface Vlan10
```

```
ip address 172.16.2.3 255.255.255.0
noip route-cache
standby 10 ip 172.16.2.1
standby 10 timers 2 4
standby 10 priority 90
!
```

All the ports shown above belong to Vlan 10.

Configuration for NAT Router (Cisco 2600)

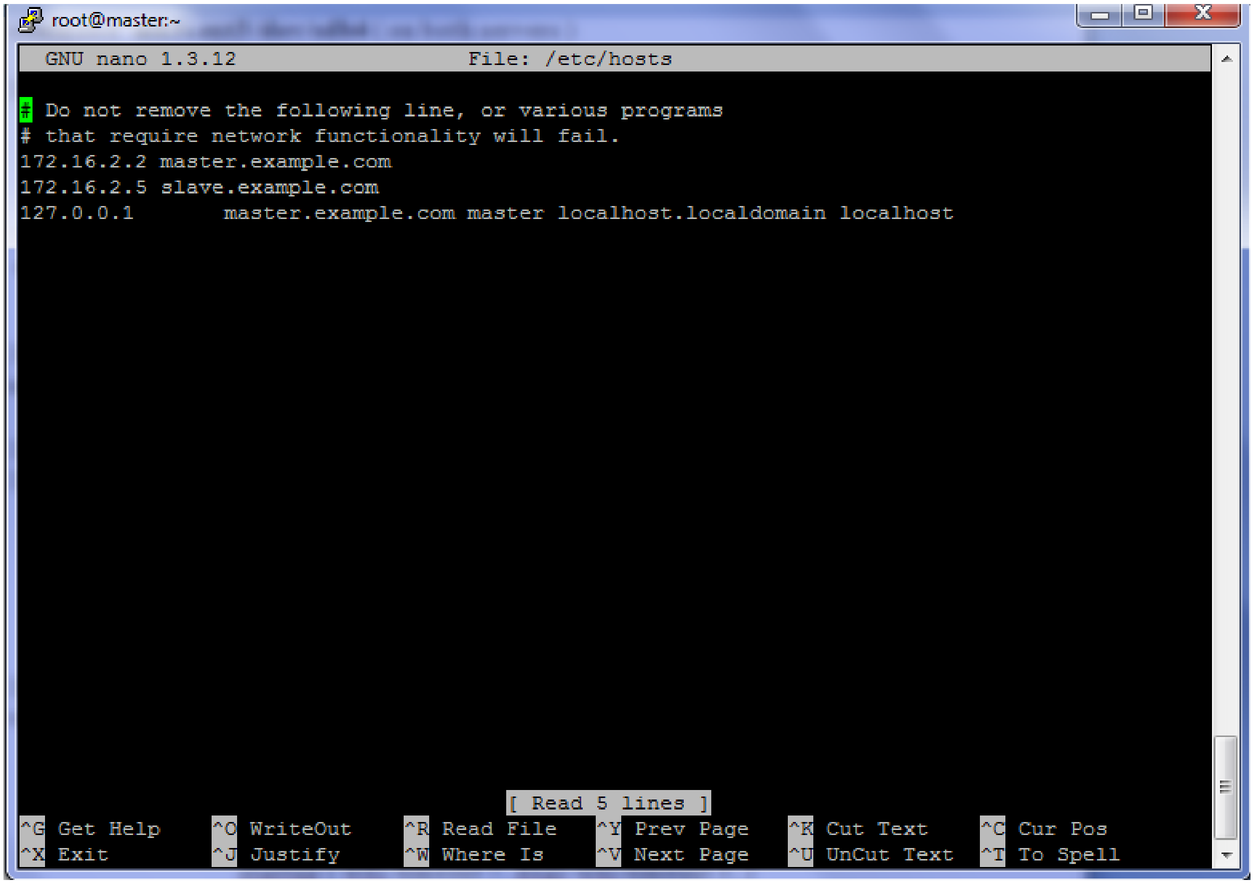
```
hostnameNAT_Router
!
boot-start-marker
boot-end-marker
!
!
memory-sizeiomem 10
noaaa new-model
ip subnet-zero
!
!
!
ipcef
!
!
!
!
!
interface FastEthernet0/0
ip address 192.168.10.99 255.255.255.0
ipnat outside
duplex auto
speed auto
!
interface Serial0/0
noip address
shutdown
!
interface FastEthernet0/1
```

```
ip address 172.16.2.20 255.255.255.0
ipnat inside
duplex auto
speed auto
!
interface Serial0/1
noip address
shutdown
!
ipnat inside source list 1 interface FastEthernet0/0 overload
ip http server
ip classless
!
!
access-list 1 permit 172.16.2.0 0.0.0.255
!
!
!
voice-port 1/0/0
!
voice-port 1/0/1
!
!
!
!
line con 0
line aux 0
linevty 0 4
!
!
end
```

Note: Router R1, Router R4, NAT Router, Switch SW1 and Switch SW2 are at customer end.

In this test we need to make few changes during the installation of servers.
We have to follow all the steps which are explained above except the following steps.

14. We have to edit /etc/hosts

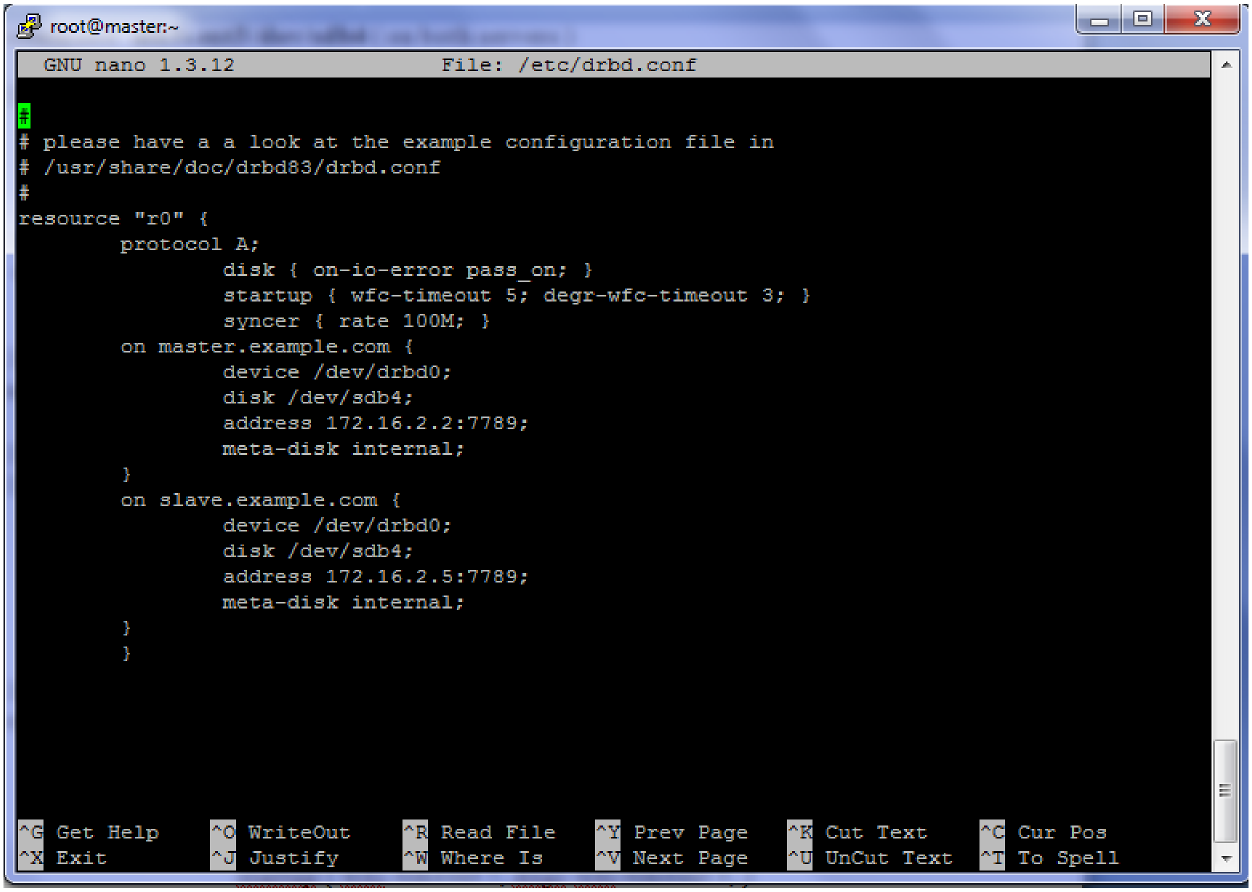


```
  GNU nano 1.3.12                  File: /etc/hosts

# Do not remove the following line, or various programs
# that require network functionality will fail.
172.16.2.2 master.example.com
172.16.2.5 slave.example.com
127.0.0.1       master.example.com master localhost.localdomain localhost

                                      [ Read 5 lines ]
^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Note: This should be same on both the servers.

Here we have defined that the server with ip address 172.16.2.2 has a hostname master.example.com and a server with ip address 172.16.2.5 has a hostname slave.example.com respectively and both are the nodes of the same cluster.

15. We need to change /etc/drbd.conf

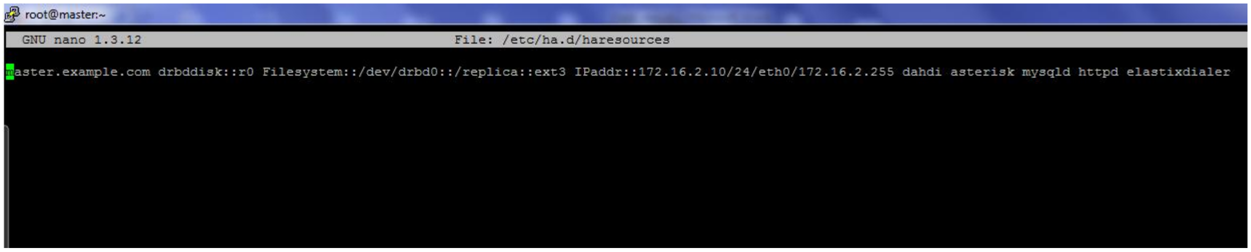


```
  GNU nano 1.3.12                File: /etc/drbd.conf

#
# please have a a look at the example configuration file in
# /usr/share/doc/drbd83/drbd.conf
#
resource "r0" {
        protocol A;
                disk { on-io-error pass_on; }
                startup { wfc-timeout 5; degr-wfc-timeout 3; }
                syncer { rate 100M; }
        on master.example.com {
                device /dev/drbd0;
                disk /dev/sdb4;
                address 172.16.2.2:7789;
                meta-disk internal;
        }
        on slave.example.com {
                device /dev/drbd0;
                disk /dev/sdb4;
                address 172.16.2.5:7789;
                meta-disk internal;
        }
        }

^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Note: This should be same on both the servers.

Here we are defining the memory space which is to be synchronized between the servers with their host names. We have also defined the synchronization rate in this file, which in our case is 100 M.

27. We have to edit /etc/ha.d/haresources



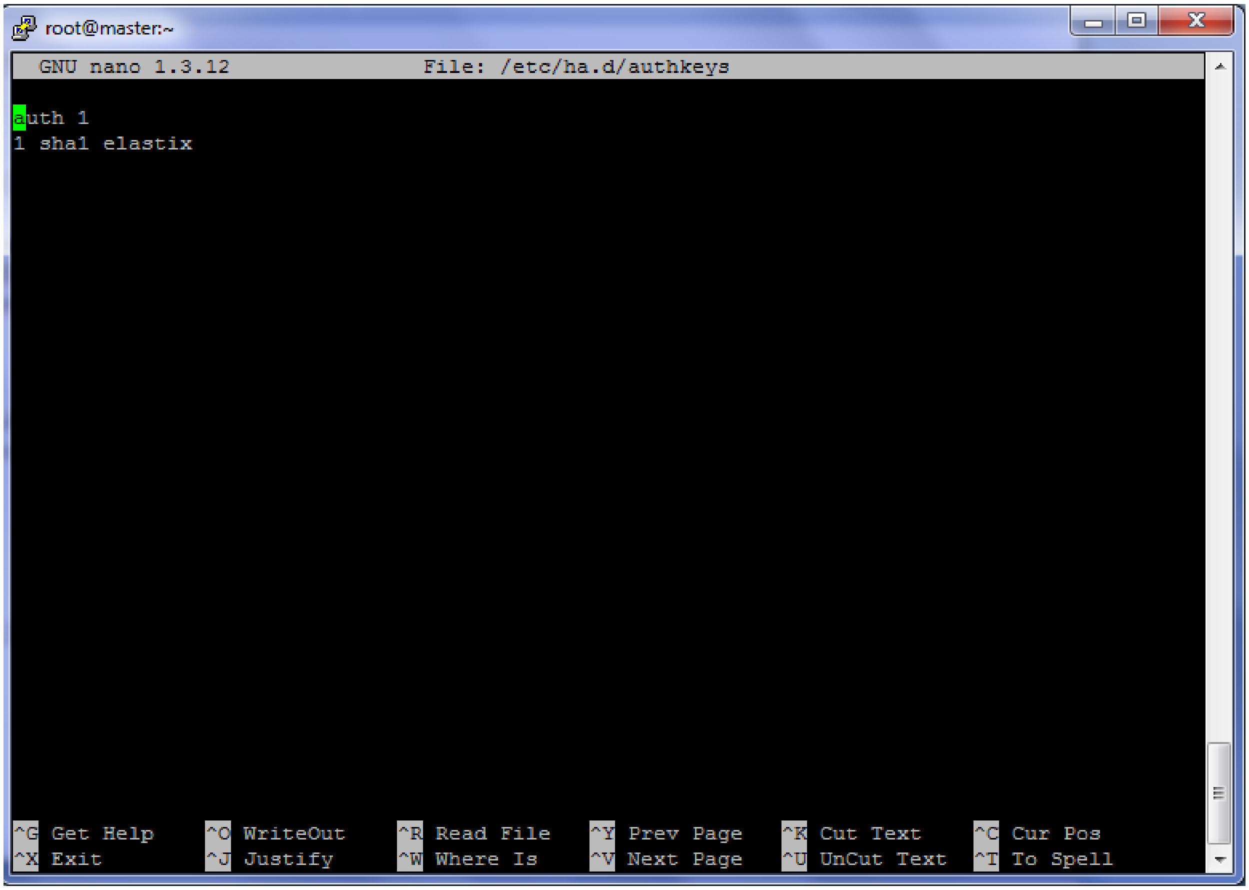
```
  GNU nano 1.3.12                File: /etc/ha.d/haresources

master.example.com drbddisk::r0 Filesystem::/dev/drbd0::/replica::ext3 IPaddr::172.16.2.10/24/eth0/172.16.2.255 dahdi asterisk mysqld httpd elastixdialer
```

Note: This should be same on both the servers.

In this file we have defined a floating ip address which is 172.16.2.10 and all the services which we want to run on both the servers.

28. In this step we will create a file that we want to have for high availability clustering, which has the authentication key defined in it.

Command: nano /etc/ha.d/authkeys



```
  GNU nano 1.3.12                File: /etc/ha.d/authkeys

auth 1
1 sha1 elastix

^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Split-Brain

If DRBD detects that both nodes are (or were at some point, while disconnected) in the primary role, it immediately tears down the replication connection. This situation is called Split-Brain. Split-Brain issue can be resolved manually as well as by running an automatic script. In our case we are going to do it manually. Following is the procedure for resolving an issue of split-brain manually.

If we get the following output, after executing drbdadm role r0 or if a server was taken out of service and then set it back on-line:

```
cs: StandAlonest:Secondary/Unknown ds:UpToDate/Unknown
orcs: StandAlonest:Primary/Unknown ds:UpToDate/Unknown
```

To resolve this issue we have to execute following:

On out dated secondary server

```
1. drbdadm secondary r0
2. drbdadm disconnect all
3. drbdadm -- --discard-my-data connect r0
```

On Primary server

```
1. drbdadm connect r0
```

Now our servers are synchronized and are working as master and slave.

We have two ip phones, one is connected at the location where master server is located, with the name of TEST1@2001 with an ip addresss 172.16.2.30 and the other phone is connected on the other end with the name of TEST2@2002 with an ip address of 172.16.2.29.

Now it's time to test the behavior of the servers.

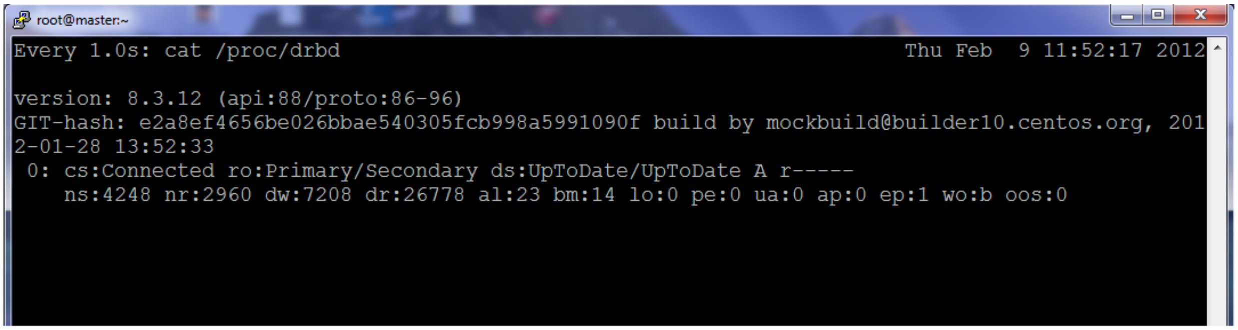
Final results

A. When both the servers are in the original state, which means heart beat is running between them and servers are fully synchronized with each other.

We can check the role and the synchronization status of the servers by executing

```
watch -n 1 cat /proc/drbd
```

Output on master server:-



```
Every 1.0s: cat /proc/drbd                                   Thu Feb  9 11:52:17 2012

version: 8.3.12 (api:88/proto:86-96)
GIT-hash: e2a8ef4656be026bbae540305fcb998a5991090f build by mockbuild@builder10.centos.org, 201
2-01-28 13:52:33
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate A r-----
    ns:4248 nr:2960 dw:7208 dr:26778 al:23 bm:14 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
```

In above screen shot we can see that both the servers are up to date and are acting normally.

Now we will make a call from TEST1@2001 to Test2@2002.

Output on server with ip address 172.16.2.10, which is our virtual ip.

```
[root@master ~]# asterisk -rvvvvvvvvvvvvvvvvvvvvv
Asterisk 1.8.7.0, Copyright (C) 1999 - 2011 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
========================================================================================
  == Parsing '/etc/asterisk/asterisk.conf':   == Found
  == Parsing '/etc/asterisk/extconfig.conf':   == Found
Connected to Asterisk 1.8.7.0 currently running on master (pid = 3982)
Verbosity is at least 21
master*CLI>
[2]+  Stopped                 asterisk -rvvvvvvvvvvvvvvvvvvvvv
[root@master ~]# clear
[root@master ~]# asterisk -rvvvvvvvvvvvvvvvvvvvvv
Asterisk 1.8.7.0, Copyright (C) 1999 - 2011 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
```

```
================================================================================
  == Parsing '/etc/asterisk/asterisk.conf':   == Found
  == Parsing '/etc/asterisk/extconfig.conf':   == Found
Connected to Asterisk 1.8.7.0 currently running on master (pid = 3982)
Verbosity is at least 21
  == Using SIP RTP TOS bits 184
  == Using SIP RTP CoS mark 5
    -- Executing [2002@from-internal:1] Macro("SIP/2001-0000000e", "exten-
vm,2002,2002") in new stack
    -- Executing [s@macro-exten-vm:1] Macro("SIP/2001-0000000e", "user-callerid,") in
new stack
    -- Executing [s@macro-user-callerid:1] Set("SIP/2001-0000000e", "AMPUSER=2001")
in new stack
    -- Executing [s@macro-user-callerid:2] GotoIf("SIP/2001-0000000e", "0?report") in
new stack
    -- Executing [s@macro-user-callerid:3] ExecIf("SIP/2001-0000000e",
"1?Set(REALCALLERIDNUM=2001)") in new stack
    -- Executing [s@macro-user-callerid:4] Set("SIP/2001-0000000e", "AMPUSER=2001")
in new stack
    -- Executing [s@macro-user-callerid:5] Set("SIP/2001-0000000e",
"AMPUSERCIDNAME=Test1") in new stack
    -- Executing [s@macro-user-callerid:6] GotoIf("SIP/2001-0000000e", "0?report") in
new stack
    -- Executing [s@macro-user-callerid:7] Set("SIP/2001-0000000e",
"AMPUSERCID=2001") in new stack
    -- Executing [s@macro-user-callerid:8] Set("SIP/2001-0000000e",
"CALLERID(all)="Test1" <2001>") in new stack
    -- Executing [s@macro-user-callerid:9] ExecIf("SIP/2001-0000000e",
"0?Set(CHANNEL(language)=)") in new stack
    -- Executing [s@macro-user-callerid:10] GotoIf("SIP/2001-0000000e", "0?continue") in
new stack
    -- Executing [s@macro-user-callerid:11] Set("SIP/2001-0000000e", "__TTL=64") in new
stack
    -- Executing [s@macro-user-callerid:12] GotoIf("SIP/2001-0000000e", "1?continue") in
new stack
    -- Goto (macro-user-callerid,s,19)
    -- Executing [s@macro-user-callerid:19] Set("SIP/2001-0000000e",
"CALLERID(number)=2001") in new stack
    -- Executing [s@macro-user-callerid:20] Set("SIP/2001-0000000e",
"CALLERID(name)=Test1") in new stack
    -- Executing [s@macro-user-callerid:21] NoOp("SIP/2001-0000000e", "Using CallerID
"Test1" <2001>") in new stack
```

```
    -- Executing [s@macro-exten-vm:2] Set("SIP/2001-0000000e", "RingGroupMethod=none") in new stack
    -- Executing [s@macro-exten-vm:3] Set("SIP/2001-0000000e", "VMBOX=2002") in new stack
    -- Executing [s@macro-exten-vm:4] Set("SIP/2001-0000000e", "__EXTTOCALL=2002") in new stack
    -- Executing [s@macro-exten-vm:5] Set("SIP/2001-0000000e", "CFUEXT=") in new stack
    -- Executing [s@macro-exten-vm:6] Set("SIP/2001-0000000e", "CFBEXT=") in new stack
    -- Executing [s@macro-exten-vm:7] Set("SIP/2001-0000000e", "RT=15") in new stack
    -- Executing [s@macro-exten-vm:8] Macro("SIP/2001-0000000e", "record-enable,2002,IN") in new stack
    -- Executing [s@macro-record-enable:1] GotoIf("SIP/2001-0000000e", "1?check") in new stack
    -- Goto (macro-record-enable,s,4)
    -- Executing [s@macro-record-enable:4] ExecIf("SIP/2001-0000000e", "0?MacroExit()") in new stack
    -- Executing [s@macro-record-enable:5] GotoIf("SIP/2001-0000000e", "0?Group:OUT") in new stack
    -- Goto (macro-record-enable,s,15)
    -- Executing [s@macro-record-enable:15] GotoIf("SIP/2001-0000000e", "1?IN") in new stack
    -- Goto (macro-record-enable,s,20)
    -- Executing [s@macro-record-enable:20] ExecIf("SIP/2001-0000000e", "1?MacroExit()") in new stack
    -- Executing [s@macro-exten-vm:9] Macro("SIP/2001-0000000e", "dial-one,15,tr,2002") in new stack
    -- Executing [s@macro-dial-one:1] Set("SIP/2001-0000000e", "DEXTEN=2002") in new stack
    -- Executing [s@macro-dial-one:2] Set("SIP/2001-0000000e", "DIALSTATUS_CW=") in new stack
    -- Executing [s@macro-dial-one:3] GosubIf("SIP/2001-0000000e", "0?screen,1") in new stack
    -- Executing [s@macro-dial-one:4] GosubIf("SIP/2001-0000000e", "0?cf,1") in new stack
    -- Executing [s@macro-dial-one:5] GotoIf("SIP/2001-0000000e", "1?skip1") in new stack
    -- Goto (macro-dial-one,s,8)
    -- Executing [s@macro-dial-one:8] GotoIf("SIP/2001-0000000e", "0?nodial") in new stack
    -- Executing [s@macro-dial-one:9] GotoIf("SIP/2001-0000000e", "0?continue") in new
stack
    -- Executing [s@macro-dial-one:10] Set("SIP/2001-0000000e", "EXTHASCW=") in new
stack
    -- Executing [s@macro-dial-one:11] GotoIf("SIP/2001-0000000e",
"1?next1:cwinusebusy") in new stack
    -- Goto (macro-dial-one,s,12)
    -- Executing [s@macro-dial-one:12] GotoIf("SIP/2001-0000000e", "0?docfu:skip3") in
new stack
    -- Goto (macro-dial-one,s,16)
    -- Executing [s@macro-dial-one:16] GotoIf("SIP/2001-0000000e", "1?next2:continue")
in new stack
    -- Goto (macro-dial-one,s,17)
    -- Executing [s@macro-dial-one:17] GotoIf("SIP/2001-0000000e", "1?continue") in
new stack
    -- Goto (macro-dial-one,s,25)
    -- Executing [s@macro-dial-one:25] GotoIf("SIP/2001-0000000e", "0?nodial") in new
stack
    -- Executing [s@macro-dial-one:26] GosubIf("SIP/2001-0000000e",
"1?dstring,1:dlocal,1") in new stack
    -- Executing [dstring@macro-dial-one:1] Set("SIP/2001-0000000e", "DSTRING=") in
new stack
    -- Executing [dstring@macro-dial-one:2] Set("SIP/2001-0000000e", "DEVICES=2002")
in new stack
    -- Executing [dstring@macro-dial-one:3] ExecIf("SIP/2001-0000000e", "0?Return()") in
new stack
    -- Executing [dstring@macro-dial-one:4] ExecIf("SIP/2001-0000000e",
"0?Set(DEVICES=002)") in new stack
    -- Executing [dstring@macro-dial-one:5] Set("SIP/2001-0000000e", "LOOPCNT=1") in
new stack
    -- Executing [dstring@macro-dial-one:6] Set("SIP/2001-0000000e", "ITER=1") in new
stack
    -- Executing [dstring@macro-dial-one:7] Set("SIP/2001-0000000e",
"THISDIAL=SIP/2002") in new stack
    -- Executing [dstring@macro-dial-one:8] GosubIf("SIP/2001-0000000e",
"1?zap2dahdi,1") in new stack
    -- Executing [zap2dahdi@macro-dial-one:1] ExecIf("SIP/2001-0000000e",
"0?Return()") in new stack
    -- Executing [zap2dahdi@macro-dial-one:2] Set("SIP/2001-0000000e", "NEWDIAL=") in
new stack
    -- Executing [zap2dahdi@macro-dial-one:3] Set("SIP/2001-0000000e",
"LOOPCNT2=1") in new stack
```

```
   -- Executing [zap2dahdi@macro-dial-one:4] Set("SIP/2001-0000000e", "ITER2=1") in
new stack
   -- Executing [zap2dahdi@macro-dial-one:5] Set("SIP/2001-0000000e",
"THISPART2=SIP/2002") in new stack
   -- Executing [zap2dahdi@macro-dial-one:6] ExecIf("SIP/2001-0000000e",
"0?Set(THISPART2=DAHDI/2002)") in new stack
   -- Executing [zap2dahdi@macro-dial-one:7] Set("SIP/2001-0000000e",
"NEWDIAL=SIP/2002&") in new stack
   -- Executing [zap2dahdi@macro-dial-one:8] Set("SIP/2001-0000000e", "ITER2=2") in
new stack
   -- Executing [zap2dahdi@macro-dial-one:9] GotoIf("SIP/2001-0000000e", "0?begin2")
in new stack
   -- Executing [zap2dahdi@macro-dial-one:10] Set("SIP/2001-0000000e",
"THISDIAL=SIP/2002") in new stack
   -- Executing [zap2dahdi@macro-dial-one:11] Return("SIP/2001-0000000e", "") in new
stack
   -- Executing [dstring@macro-dial-one:9] Set("SIP/2001-0000000e",
"DSTRING=SIP/2002&") in new stack
   -- Executing [dstring@macro-dial-one:10] Set("SIP/2001-0000000e", "ITER=2") in new
stack
   -- Executing [dstring@macro-dial-one:11] GotoIf("SIP/2001-0000000e", "0?begin") in
new stack
   -- Executing [dstring@macro-dial-one:12] Set("SIP/2001-0000000e",
"DSTRING=SIP/2002") in new stack
   -- Executing [dstring@macro-dial-one:13] Return("SIP/2001-0000000e", "") in new
stack
   -- Executing [s@macro-dial-one:27] GotoIf("SIP/2001-0000000e", "0?nodial") in new
stack
   -- Executing [s@macro-dial-one:28] GotoIf("SIP/2001-0000000e", "1?skiptrace") in
new stack
   -- Goto (macro-dial-one,s,30)
   -- Executing [s@macro-dial-one:30] Set("SIP/2001-0000000e", "D_OPTIONS=tr") in
new stack
   -- Executing [s@macro-dial-one:31] ExecIf("SIP/2001-0000000e",
"0?SIPAddHeader(Alert-Info: )") in new stack
   -- Executing [s@macro-dial-one:32] ExecIf("SIP/2001-0000000e", "0?SIPAddHeader()")
in new stack
   -- Executing [s@macro-dial-one:33] ExecIf("SIP/2001-0000000e",
"0?Set(CHANNEL(musicclass)=)") in new stack
   -- Executing [s@macro-dial-one:34] GosubIf("SIP/2001-0000000e", "0?qwait,1") in
new stack
```

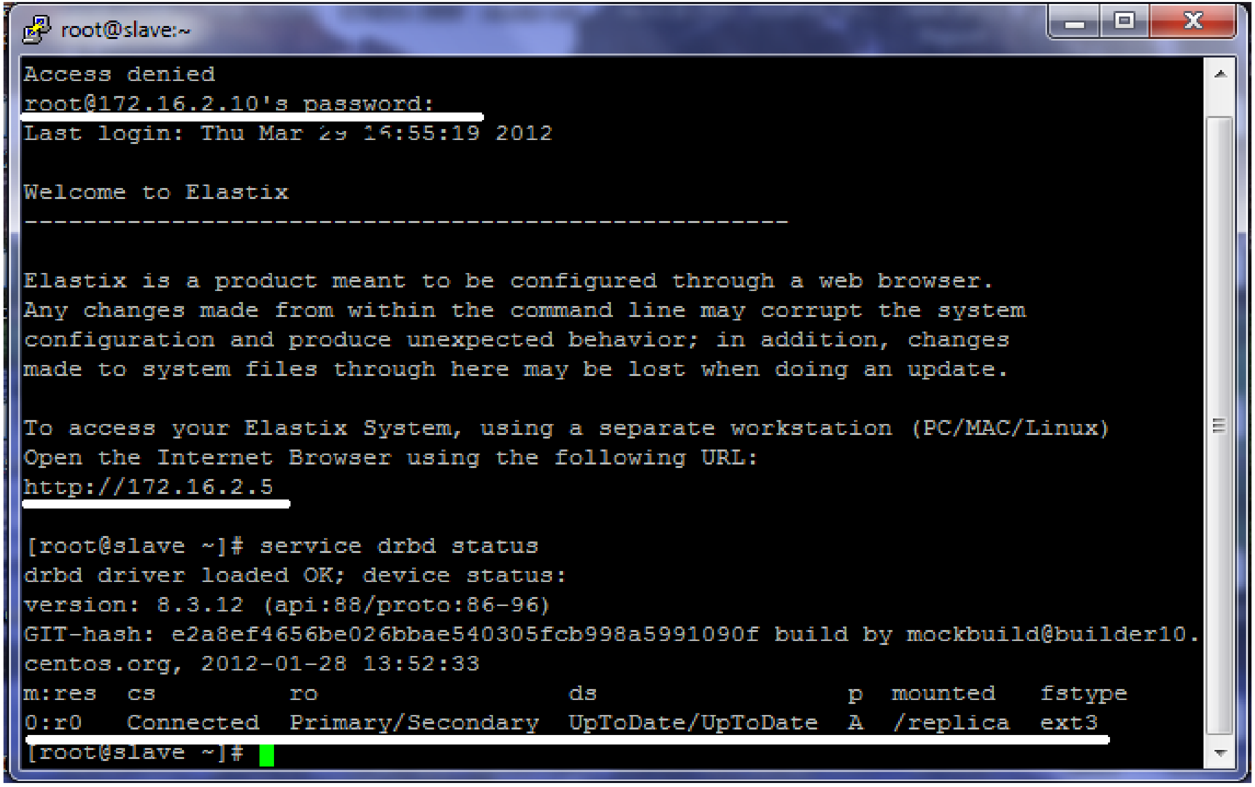
```
    -- Executing [s@macro-dial-one:35] Set("SIP/2001-0000000e", "__CWIGNORE=") in
new stack
    -- Executing [s@macro-dial-one:36] Set("SIP/2001-0000000e", "__KEEPCID=TRUE") in
new stack
    -- Executing [s@macro-dial-one:37] Dial("SIP/2001-0000000e", "SIP/2002,15,tr") in
new stack
  == Using SIP RTP TOS bits 184
  == Using SIP RTP CoS mark 5
-- Called SIP/2002
    -- SIP/2002-0000000f is ringing
    -- SIP/2002-0000000f is making progress passing it to SIP/2001-0000000e
    -- SIP/2002-0000000f is ringing
    -- SIP/2002-0000000f is making progress passing it to SIP/2001-0000000e
    -- SIP/2002-0000000f answered SIP/2001-0000000e
    -- Remote UNIX connection
    -- Remote UNIX connection disconnected
    -- Executing [h@macro-dial-one:1] Macro("SIP/2001-0000000e", "hangupcall,") in new
stack
    -- Executing [s@macro-hangupcall:1] GotoIf("SIP/2001-0000000e",
"1?endmixmoncheck") in new stack
    -- Goto (macro-hangupcall,s,9)
    -- Executing [s@macro-hangupcall:9] NoOp("SIP/2001-0000000e", "End of MIXMON
check") in new stack
    -- Executing [s@macro-hangupcall:10] GotoIf("SIP/2001-0000000e",
"1?nomeetmemon") in new stack
    -- Goto (macro-hangupcall,s,15)
    -- Executing [s@macro-hangupcall:15] NoOp("SIP/2001-0000000e",
"MEETME_RECORDINGFILE=") in new stack
    -- Executing [s@macro-hangupcall:16] GotoIf("SIP/2001-0000000e", "1?noautomon")
in new stack
    -- Goto (macro-hangupcall,s,18)
    -- Executing [s@macro-hangupcall:18] NoOp("SIP/2001-0000000e",
"TOUCH_MONITOR_OUTPUT=") in new stack
    -- Executing [s@macro-hangupcall:19] GotoIf("SIP/2001-0000000e", "1?noautomon2")
in new stack
    -- Goto (macro-hangupcall,s,25)
    -- Executing [s@macro-hangupcall:25] NoOp("SIP/2001-0000000e",
"MONITOR_FILENAME=") in new stack
    -- Executing [s@macro-hangupcall:26] GotoIf("SIP/2001-0000000e", "1?skiprg") in
new stack
    -- Goto (macro-hangupcall,s,29)
    -- Executing [s@macro-hangupcall:29] GotoIf("SIP/2001-0000000e", "1?skipblkvm") in new stack
    -- Goto (macro-hangupcall,s,32)
    -- Executing [s@macro-hangupcall:32] GotoIf("SIP/2001-0000000e", "1?theend") in new stack
    -- Goto (macro-hangupcall,s,34)
    -- Executing [s@macro-hangupcall:34] Hangup("SIP/2001-0000000e", "") in new stack
  == Spawn extension (macro-hangupcall, s, 34) exited non-zero on 'SIP/2001-0000000e' in macro 'hangupcall'
  == Spawn extension (macro-dial-one, h, 1) exited non-zero on 'SIP/2001-0000000e'
  == Spawn extension (macro-dial-one, s, 37) exited non-zero on 'SIP/2001-0000000e' in macro 'dial-one'
  == Spawn extension (macro-exten-vm, s, 9) exited non-zero on 'SIP/2001-0000000e' in macro 'exten-vm'
  == Spawn extension (from-internal, 2002, 1) exited non-zero on 'SIP/2001-0000000e'
```

The highlighted lines show that the call is in progress.

B. Now it’s time to check what will happen when heartbeat stops on master server.

Execute this on master server: Service heartbeat stop

Output on the server with ip address 172.16.2.10 (Floating ip address) :-



```
Access denied
root@172.16.2.10's password:
Last login: Thu Mar 29 16:55:19 2012

Welcome to Elastix
----------------------------------------------------

Elastix is a product meant to be configured through a web browser.
Any changes made from within the command line may corrupt the system
configuration and produce unexpected behavior; in addition, changes
made to system files through here may be lost when doing an update.

To access your Elastix System, using a separate workstation (PC/MAC/Linux)
Open the Internet Browser using the following URL:
http://172.16.2.5

[root@slave ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.3.12 (api:88/proto:86-96)
GIT-hash: e2a8ef4656be026bbae540305fcb998a5991090f build by mockbuild@builder10.
centos.org, 2012-01-28 13:52:33
m:res  cs         ro                 ds                 p  mounted   fstype
0:r0   Connected  Primary/Secondary  UpToDate/UpToDate  A  /replica  ext3
[root@slave ~]#
```

In the above screen shot we are logged in to the server through our floating ip address which is 172.16.2.10. As the heartbeat service is stopped manually between the two servers, drbd status changes. Our secondary server automatically changes to primary server. Now our server with ip address 172.16.2.5 is acting like a primary server and the server with the ip address 172.16.2.2 is a secondary server. All the phones are registered with the floating ip address.

We will again make a call from TEST1@2001 to Test2@2002.

Output:

```
[root@slave ~]# asterisk -rvvvvvvvvvvvvvvvvv
Asterisk 1.8.7.0, Copyright (C) 1999 - 2011 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
  == Parsing '/etc/asterisk/asterisk.conf':   == Found
  == Parsing '/etc/asterisk/extconfig.conf':   == Found
Connected to Asterisk 1.8.7.0 currently running on slave (pid = 4021)
Verbosity is at least 17
    -- Executing [h@macro-dial-one:1] Macro("SIP/2001-00000000", "hangupcall,") in new
stack
    -- Executing [s@macro-hangupcall:1] GotoIf("SIP/2001-00000000",
"1?endmixmoncheck") in new stack
    -- Goto (macro-hangupcall,s,9)
    -- Executing [s@macro-hangupcall:9] NoOp("SIP/2001-00000000", "End of MIXMON
check") in new stack
    -- Executing [s@macro-hangupcall:10] GotoIf("SIP/2001-00000000",
"1?nomeetmemon") in new stack
    -- Goto (macro-hangupcall,s,15)
    -- Executing [s@macro-hangupcall:15] NoOp("SIP/2001-00000000",
"MEETME_RECORDINGFILE=") in new stack
    -- Executing [s@macro-hangupcall:16] GotoIf("SIP/2001-00000000", "1?noautomon")
in new stack
    -- Goto (macro-hangupcall,s,18)
    -- Executing [s@macro-hangupcall:18] NoOp("SIP/2001-00000000",
"TOUCH_MONITOR_OUTPUT=") in new stack
```

```
    -- Executing [s@macro-hangupcall:19] GotoIf("SIP/2001-00000000", "1?noautomon2")
in new stack
    -- Goto (macro-hangupcall,s,25)
    -- Executing [s@macro-hangupcall:25] NoOp("SIP/2001-00000000",
"MONITOR_FILENAME=") in new stack
    -- Executing [s@macro-hangupcall:26] GotoIf("SIP/2001-00000000", "1?skiprg") in
new stack
    -- Goto (macro-hangupcall,s,29)
    -- Executing [s@macro-hangupcall:29] GotoIf("SIP/2001-00000000", "1?skipblkvm") in
new stack
    -- Goto (macro-hangupcall,s,32)
    -- Executing [s@macro-hangupcall:32] GotoIf("SIP/2001-00000000", "1?theend") in
new stack
    -- Goto (macro-hangupcall,s,34)
    -- Executing [s@macro-hangupcall:34] Hangup("SIP/2001-00000000", "") in new stack
  == Spawn extension (macro-hangupcall, s, 34) exited non-zero on 'SIP/2001-00000000'
in macro 'hangupcall'
  == Spawn extension (macro-dial-one, h, 1) exited non-zero on 'SIP/2001-00000000'
  == Spawn extension (macro-dial-one, s, 37) exited non-zero on 'SIP/2001-00000000' in
macro 'dial-one'
  == Spawn extension (macro-exten-vm, s, 9) exited non-zero on 'SIP/2001-00000000' in
macro 'exten-vm'
  == Spawn extension (from-internal, 2002, 1) exited non-zero on 'SIP/2001-00000000'
  == Using SIP RTP TOS bits 184
  == Using SIP RTP CoS mark 5
    -- Executing [2002@from-internal:1] Macro("SIP/2001-00000002", "exten-
vm,2002,2002") in new stack
    -- Executing [s@macro-exten-vm:1] Macro("SIP/2001-00000002", "user-callerid,") in
new stack
    -- Executing [s@macro-user-callerid:1] Set("SIP/2001-00000002", "AMPUSER=2001")
in new stack
    -- Executing [s@macro-user-callerid:2] GotoIf("SIP/2001-00000002", "0?report") in
new stack
    -- Executing [s@macro-user-callerid:3] ExecIf("SIP/2001-00000002",
"1?Set(REALCALLERIDNUM=2001)") in new stack
    -- Executing [s@macro-user-callerid:4] Set("SIP/2001-00000002", "AMPUSER=2001")
in new stack
    -- Executing [s@macro-user-callerid:5] Set("SIP/2001-00000002",
"AMPUSERCIDNAME=Test1") in new stack
    -- Executing [s@macro-user-callerid:6] GotoIf("SIP/2001-00000002", "0?report") in
new stack
```

```
   -- Executing [s@macro-user-callerid:7] Set("SIP/2001-00000002",
"AMPUSERCID=2001") in new stack
   -- Executing [s@macro-user-callerid:8] Set("SIP/2001-00000002",
"CALLERID(all)="Test1" <2001>") in new stack
   -- Executing [s@macro-user-callerid:9] ExecIf("SIP/2001-00000002",
"0?Set(CHANNEL(language)=)") in new stack
   -- Executing [s@macro-user-callerid:10] GotoIf("SIP/2001-00000002", "0?continue") in
new stack
   -- Executing [s@macro-user-callerid:11] Set("SIP/2001-00000002", "__TTL=64") in new
stack
   -- Executing [s@macro-user-callerid:12] GotoIf("SIP/2001-00000002", "1?continue") in
new stack
   -- Goto (macro-user-callerid,s,19)
   -- Executing [s@macro-user-callerid:19] Set("SIP/2001-00000002",
"CALLERID(number)=2001") in new stack
   -- Executing [s@macro-user-callerid:20] Set("SIP/2001-00000002",
"CALLERID(name)=Test1") in new stack
   -- Executing [s@macro-user-callerid:21] NoOp("SIP/2001-00000002", "Using CallerID
"Test1" <2001>") in new stack
   -- Executing [s@macro-exten-vm:2] Set("SIP/2001-00000002",
"RingGroupMethod=none") in new stack
   -- Executing [s@macro-exten-vm:3] Set("SIP/2001-00000002", "VMBOX=2002") in new
stack
   -- Executing [s@macro-exten-vm:4] Set("SIP/2001-00000002", "__EXTTOCALL=2002")
in new stack
   -- Executing [s@macro-exten-vm:5] Set("SIP/2001-00000002", "CFUEXT=") in new
stack
   -- Executing [s@macro-exten-vm:6] Set("SIP/2001-00000002", "CFBEXT=") in new
stack
   -- Executing [s@macro-exten-vm:7] Set("SIP/2001-00000002", "RT=15") in new stack
   -- Executing [s@macro-exten-vm:8] Macro("SIP/2001-00000002", "record-
enable,2002,IN") in new stack
   -- Executing [s@macro-record-enable:1] GotoIf("SIP/2001-00000002", "1?check") in
new stack
   -- Goto (macro-record-enable,s,4)
   -- Executing [s@macro-record-enable:4] ExecIf("SIP/2001-00000002",
"0?MacroExit()") in new stack
   -- Executing [s@macro-record-enable:5] GotoIf("SIP/2001-00000002",
"0?Group:OUT") in new stack
   -- Goto (macro-record-enable,s,15)
   -- Executing [s@macro-record-enable:15] GotoIf("SIP/2001-00000002", "1?IN") in new
stack
```

```
    -- Goto (macro-record-enable,s,20)
    -- Executing [s@macro-record-enable:20] ExecIf("SIP/2001-00000002",
"1?MacroExit()") in new stack
    -- Executing [s@macro-exten-vm:9] Macro("SIP/2001-00000002", "dial-
one,15,tr,2002") in new stack
    -- Executing [s@macro-dial-one:1] Set("SIP/2001-00000002", "DEXTEN=2002") in new
stack
    -- Executing [s@macro-dial-one:2] Set("SIP/2001-00000002", "DIALSTATUS_CW=") in
new stack
    -- Executing [s@macro-dial-one:3] GosubIf("SIP/2001-00000002", "0?screen,1") in
new stack
    -- Executing [s@macro-dial-one:4] GosubIf("SIP/2001-00000002", "0?cf,1") in new
stack
    -- Executing [s@macro-dial-one:5] GotoIf("SIP/2001-00000002", "1?skip1") in new
stack
    -- Goto (macro-dial-one,s,8)
    -- Executing [s@macro-dial-one:8] GotoIf("SIP/2001-00000002", "0?nodial") in new
stack
    -- Executing [s@macro-dial-one:9] GotoIf("SIP/2001-00000002", "0?continue") in new
stack
    -- Executing [s@macro-dial-one:10] Set("SIP/2001-00000002", "EXTHASCW=") in new
stack
    -- Executing [s@macro-dial-one:11] GotoIf("SIP/2001-00000002",
"1?next1:cwinusebusy") in new stack
    -- Goto (macro-dial-one,s,12)
    -- Executing [s@macro-dial-one:12] GotoIf("SIP/2001-00000002", "0?docfu:skip3") in
new stack
    -- Goto (macro-dial-one,s,16)
    -- Executing [s@macro-dial-one:16] GotoIf("SIP/2001-00000002", "1?next2:continue")
in new stack
    -- Goto (macro-dial-one,s,17)
    -- Executing [s@macro-dial-one:17] GotoIf("SIP/2001-00000002", "1?continue") in
new stack
    -- Goto (macro-dial-one,s,25)
    -- Executing [s@macro-dial-one:25] GotoIf("SIP/2001-00000002", "0?nodial") in new
stack
    -- Executing [s@macro-dial-one:26] GosubIf("SIP/2001-00000002",
"1?dstring,1:dlocal,1") in new stack
    -- Executing [dstring@macro-dial-one:1] Set("SIP/2001-00000002", "DSTRING=") in
new stack
    -- Executing [dstring@macro-dial-one:2] Set("SIP/2001-00000002", "DEVICES=2002")
in new stack
```

```
    -- Executing [dstring@macro-dial-one:3] ExecIf("SIP/2001-00000002", "0?Return()") in
new stack
    -- Executing [dstring@macro-dial-one:4] ExecIf("SIP/2001-00000002",
"0?Set(DEVICES=002)") in new stack
    -- Executing [dstring@macro-dial-one:5] Set("SIP/2001-00000002", "LOOPCNT=1") in
new stack
    -- Executing [dstring@macro-dial-one:6] Set("SIP/2001-00000002", "ITER=1") in new
stack
    -- Executing [dstring@macro-dial-one:7] Set("SIP/2001-00000002",
"THISDIAL=SIP/2002") in new stack
    -- Executing [dstring@macro-dial-one:8] GosubIf("SIP/2001-00000002",
"1?zap2dahdi,1") in new stack
    -- Executing [zap2dahdi@macro-dial-one:1] ExecIf("SIP/2001-00000002",
"0?Return()") in new stack
    -- Executing [zap2dahdi@macro-dial-one:2] Set("SIP/2001-00000002", "NEWDIAL=") in
new stack
    -- Executing [zap2dahdi@macro-dial-one:3] Set("SIP/2001-00000002",
"LOOPCNT2=1") in new stack
    -- Executing [zap2dahdi@macro-dial-one:4] Set("SIP/2001-00000002", "ITER2=1") in
new stack
    -- Executing [zap2dahdi@macro-dial-one:5] Set("SIP/2001-00000002",
"THISPART2=SIP/2002") in new stack
    -- Executing [zap2dahdi@macro-dial-one:6] ExecIf("SIP/2001-00000002",
"0?Set(THISPART2=DAHDI/2002)") in new stack
    -- Executing [zap2dahdi@macro-dial-one:7] Set("SIP/2001-00000002",
"NEWDIAL=SIP/2002&") in new stack
    -- Executing [zap2dahdi@macro-dial-one:8] Set("SIP/2001-00000002", "ITER2=2") in
new stack
    -- Executing [zap2dahdi@macro-dial-one:9] GotoIf("SIP/2001-00000002", "0?begin2")
in new stack
    -- Executing [zap2dahdi@macro-dial-one:10] Set("SIP/2001-00000002",
"THISDIAL=SIP/2002") in new stack
    -- Executing [zap2dahdi@macro-dial-one:11] Return("SIP/2001-00000002", "") in new
stack
    -- Executing [dstring@macro-dial-one:9] Set("SIP/2001-00000002",
"DSTRING=SIP/2002&") in new stack
    -- Executing [dstring@macro-dial-one:10] Set("SIP/2001-00000002", "ITER=2") in new
stack
    -- Executing [dstring@macro-dial-one:11] GotoIf("SIP/2001-00000002", "0?begin") in
new stack
    -- Executing [dstring@macro-dial-one:12] Set("SIP/2001-00000002",
"DSTRING=SIP/2002") in new stack
```

```
    -- Executing [dstring@macro-dial-one:13] Return("SIP/2001-00000002", "") in new
stack
    -- Executing [s@macro-dial-one:27] GotoIf("SIP/2001-00000002", "0?nodial") in new
stack
    -- Executing [s@macro-dial-one:28] GotoIf("SIP/2001-00000002", "1?skiptrace") in
new stack
    -- Goto (macro-dial-one,s,30)
    -- Executing [s@macro-dial-one:30] Set("SIP/2001-00000002", "D_OPTIONS=tr") in
new stack
    -- Executing [s@macro-dial-one:31] ExecIf("SIP/2001-00000002",
"0?SIPAddHeader(Alert-Info: )") in new stack
    -- Executing [s@macro-dial-one:32] ExecIf("SIP/2001-00000002", "0?SIPAddHeader()")
in new stack
    -- Executing [s@macro-dial-one:33] ExecIf("SIP/2001-00000002",
"0?Set(CHANNEL(musicclass)=)") in new stack
    -- Executing [s@macro-dial-one:34] GosubIf("SIP/2001-00000002", "0?qwait,1") in
new stack
    -- Executing [s@macro-dial-one:35] Set("SIP/2001-00000002", "__CWIGNORE=") in
new stack
    -- Executing [s@macro-dial-one:36] Set("SIP/2001-00000002", "__KEEPCID=TRUE") in
new stack
    -- Executing [s@macro-dial-one:37] Dial("SIP/2001-00000002", "SIP/2002,15,tr") in
new stack
  == Using SIP RTP TOS bits 184
  == Using SIP RTP CoS mark 5
    -- Called SIP/2002
    -- SIP/2002-00000003 is ringing
    -- SIP/2002-00000003 is making progress passing it to SIP/2001-00000002
    -- SIP/2002-00000003 is ringing
    -- SIP/2002-00000003 is making progress passing it to SIP/2001-00000002
    -- SIP/2002-00000003 answered SIP/2001-00000002
    -- Executing [h@macro-dial-one:1] Macro("SIP/2001-00000002", "hangupcall,") in new
stack
    -- Executing [s@macro-hangupcall:1] GotoIf("SIP/2001-00000002",
"1?endmixmoncheck") in new stack
    -- Goto (macro-hangupcall,s,9)
    -- Executing [s@macro-hangupcall:9] NoOp("SIP/2001-00000002", "End of MIXMON
check") in new stack
    -- Executing [s@macro-hangupcall:10] GotoIf("SIP/2001-00000002",
"1?nomeetmemon") in new stack
    -- Goto (macro-hangupcall,s,15)
    -- Executing [s@macro-hangupcall:15] NoOp("SIP/2001-00000002",
"MEETME_RECORDINGFILE=") in new stack
    -- Executing [s@macro-hangupcall:16] GotoIf("SIP/2001-00000002", "1?noautomon")
in new stack
    -- Goto (macro-hangupcall,s,18)
    -- Executing [s@macro-hangupcall:18] NoOp("SIP/2001-00000002",
"TOUCH_MONITOR_OUTPUT=") in new stack
    -- Executing [s@macro-hangupcall:19] GotoIf("SIP/2001-00000002", "1?noautomon2")
in new stack
    -- Goto (macro-hangupcall,s,25)
    -- Executing [s@macro-hangupcall:25] NoOp("SIP/2001-00000002",
"MONITOR_FILENAME=") in new stack
    -- Executing [s@macro-hangupcall:26] GotoIf("SIP/2001-00000002", "1?skiprg") in
new stack
    -- Goto (macro-hangupcall,s,29)
    -- Executing [s@macro-hangupcall:29] GotoIf("SIP/2001-00000002", "1?skipblkvm") in
new stack
    -- Goto (macro-hangupcall,s,32)
    -- Executing [s@macro-hangupcall:32] GotoIf("SIP/2001-00000002", "1?theend") in
new stack
    -- Goto (macro-hangupcall,s,34)
    -- Executing [s@macro-hangupcall:34] Hangup("SIP/2001-00000002", "") in new stack
  == Spawn extension (macro-hangupcall, s, 34) exited non-zero on 'SIP/2001-00000002'
in macro 'hangupcall'
  == Spawn extension (macro-dial-one, h, 1) exited non-zero on 'SIP/2001-00000002'
  == Spawn extension (macro-dial-one, s, 37) exited non-zero on 'SIP/2001-00000002' in
macro 'dial-one'
  == Spawn extension (macro-exten-vm, s, 9) exited non-zero on 'SIP/2001-00000002' in
macro 'exten-vm'
  == Spawn extension (from-internal, 2002, 1) exited non-zero on 'SIP/2001-00000002'
```

The highlighted lines show that the call is in progress.

Conclusion

Both the tests show that redundancy for the Elastix servers is feasible over the wide area network, but the only challenge was to keep the same network on the both servers. No matter where they are located. We can use L2TP or VPLS for that, depending on our requirements.

Challenges and difficulties

- In this project the biggest challenge was to learn about Elastix server due to the limited resources in English language. As Elastix is a new open source communication server there is not enough study material over the internet so far.

- I was new to all the technologies that I have used in this project but with the help of online tutorials, videos and books I was able to understand and implement VPLS, L2TP, High availability clustering and Heartbeat.

References

I. http://teal.gmu.edu/courses/ECE543/project/reports_2001/arveal.pdf

II. http://www.Elastix.org/

III. http://www.oracle.com/partners/en/knowledge-zone/server-storage/sun-blade-servers-knowledge-zone-143223.html

IV. http://www.cisco.com/en/US/docs/ios/12_0t/12_0t1/feature/guide/l2tpT.html

V. http://compnetworking.about.com/od/vpn/a/vpn_tunneling.htm

VI. http://teal.gmu.edu/courses/ECE543/project/reports_2001/arveal.pdf

VII. http://www.javvin.com/protocolL2TP.html

VIII. http://www.cisco.com/warp/public/cc/so/neso/vpn/unvpnst/2tpv3_ov.pdf

IX. Elastix Without Tears book by Ben Shariff

X. L2TPv3 overview and configuration (video tutorials)

XI. VPLS Configuration and overview (video tutorials)

XII. http://www.elastix.org/index.php/en/component/kunena/ (Elastix forums)